Universidad Nacional Autónoma de México

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES
ARAGON

INGENIERIA EN COMPUTACION

"DISEÑO DE UN SISTEMA EXPERTO EN ORIENTACION VOCACIONAL Y GUIA DE CARRERAS EN EL ESTADO DE MEXICO"

TESIS PROFESIONAL

QUE PARA OBTENER EL TITULO DE
INGENIERO EN COMPUTACION

PRESENTA

Gustavo César Martínez Lira

ASESOR: Ing. Amilcar Monterrosa Escobar

ENEP

ARAGON

San Juan de Aragón, Edo. de México. 1995.

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

A mi Dios, en Ti está el fundamento de mi vida, gracias por creer en mí.

A mis padres Vicente y Alicia, gracias porque han sido el mayor sostén en mi vida durante todo este tiempo, gracias por su amor incondicional, los amo.

A mis hermanas Blanca y Lety, gracias por apoyar mis estudios; aunque a veces no parezca, es una buena oportunidad para decirles que las quiero mucho.

A mis hermanos en la fe, José Antonio, Maru, Angela y Javier; los momentos que hemos vivido juntos compartiendo lo que es más importante para nosotros son mi motivación en tiempos difíciles, gracias por su amor.

A mis amigos Itsmael, Francisco y Martín, gracias por ser ejemplo de gente emprendedora que nunca se conforma con saber lo que la mayoría aprende en las aulas.

Gracias al Ing. Víctor Velasco Vega por el apoyo brindado a la realización de esta tesis; gracias a todos mis maestros, en especial al Ing. Juan de Dios González Romero porque me enseñó a estudiar al reprobar su primer examen parcial.

Y a todos los que esperaban encontrar su nombre en estas dedicatorias y no están, mil disculpas, pero gracias porque han sido ayuda a mi vida.

Indice General

Objetivos

OBJETIVO GENERAL

Obtener un Sistema Experto capaz de auxiliar a un estudiante de bachillerato en la elección de profesión, tomando en cuenta todos los factores requeridos para la generación de soluciones satisfactorias.

OBJETIVOS ESPECIFICOS

- Recopilar la teoría existente acerca de Sistemas Expertos para obtener un método de desarrollo.
- Fundamentar el sistema con información obtenida en literatura sobre Orientación Vocacional y experiencias de personas expertas en el área.
- Desarrollar de principio a fin un Sistema Experto nuevo y original.
- Obtener una nueva herramienta, de aplicación real, para los Orientadores Vocacionales.
- Implementar dentro del sistema una Guía de Carreras sobre Instituciones y carreras que se imparten en el Estado de México.

Introducción

Sin duda alguna el escuchar las palabras *Inteligencia Artificial*, nos provoca una sensación de innovación, avance tecnológico y hasta ciencia ficción. La verdad de las cosas es que ésta, es una de las ramas de la computación que más auge ha tenido en las últimas dos décadas. Hoy en día, hablar de aplicaciones de Inteligencia Artificial es más común que hace algunos años.

Sistemas Expertos es una de las ramas de la Inteligencia Artificial y aquella que logró obtener las primeras aplicaciones prácticas de la misma. El presente trabajo de tesis esta encaminado a la obtención de un Sistema Experto con aplicación en la Orientación Vocacional que contenga además una Guía de carreras de el Estado de México. Se pretende aplicar conocimientos y experiencias existentes sobre los Sistemas Expertos y la Orientación Vocacional para obtener un producto nuevo y con aplicación real. El capítulo I, **La Teoría de los Sistemas Expertos**, aborda toda la teoría existente sobre los mismos y permite obtener un método de desarrollo. El capítulo II, **La Elección Vocacional**, contiene la teoría indispensable para fundamentar el área de dominio de nuestro Sistema Experto. Por último, el capítulo III, **Desarrollo del Sistema Experto**, es la descripción del proceso de desarrollo de nuestro sistema. Al final, se encuentran tres apéndices que muestran el listado de los programas que conforman nuestro sistema, así como información que fue utilizada por el mismo.

Se obtiene, como producto final de esta tesis, el sistema llamado **Doctor Vocacional,** el cual puede ser aplicado de manera confiable a estudiantes de bachillerato para auxiliarles en el proceso de Orientación Vocacional.

Capítulo I

La Teoría de los Sistemas Expertos

El capítulo I de esta tesis contiene la teoría necesaria para desarrollar un sistema experto. Se describen las técnicas y métodos hasta hoy desarrollados en esta área de la Inteligencia Artificial.

En el subtema I.1, **Introducción a los Sistemas Expertos**, se describe lo que es un sistema experto, sus diferencias con un sistema tradicional, sus campos de aplicación y los elementos que lo componen.

El subtema I.2, **Representación del Conocimiento**, trata de manera más detallada la base de conocimiento, describe algunas formas de representación del conocimiento y la forma en que éste se manipula.

El subtema I.3 se titula **El Proceso de Inferencia**, y trata acerca de la forma en que el motor de inferencia emplea el conocimiento para generar soluciones. Se describen los métodos de inferencia principales: encadenamiento hacia adelante y encadenamiento hacia atrás, algunos métodos de búsqueda y algunos métodos para la solución de problemas.

En el subtema I.4, llamado **Los Módulos de Comunicación**, se habla sobre el módulo de comunicación con el usuario y el módulo de comunicación con el experto, se trata también de manera particular el componente explicativo.

Por último, el subtema I.5, **El Proceso de Construcción de un Sistema Experto**, se describe un método de desarrollo de un sistema experto, considerando cuáles problemas pueden resolverse con un sistema experto y los problemas más comunes en la construcción de uno.

I.1

Introducción a los Sistemas Expertos

ANTECEDENTES

Al inicio de la segunda mitad de este siglo existió un auge en diversas ciencias que trataban sobre el estudio del pensamiento humano, tales como la Psicología, la Filosofía o la Lingüística. Con el surgimiento de la computadora, nacieron ciencias como la Inteligencia Artificial y la Psicología cognitiva, cada una teniendo una visión sobre la inteligencia y la cognición. La Psicología cognitiva pensó en la computadora como una excelente herramienta para la comprensión del pensamiento humano utilizándola como un elemento de simulación del mismo; algunos llegaron a la conclusión de que tanto la computadora como el hombre pertenecían a una misma categoría de entes que procesan información.

Ambas ciencias, la Inteligencia Artificial y la Psicología cognitiva, convergen al tratar de encontrar en la computadora un elemento para la simulación del pensamiento, la primera para la solución de problemas reales por computadora y la segunda para obtener una mejor explicación del proceso intelectual. Nosotros nos enfocaremos aquí al trabajo realizado por la Inteligencia Artificial y más específicamente de la rama de los Sistemas Expertos, no dejando de recordar la influencia de las ciencias antes citadas.

Puntos de comparación

Cuando se habla de programas de computadora que intenten simular el pensamiento humano, se trata de encontrar factores que indiquen el nivel de éxito de dichos programas, de esta forma surgen algunos cuestionamientos:

I. La Teoría de los Sistemas Expertos

1) La solución obtenida por el programa de computadora, ¿será equiparable con la de un humano novato, un humano medio o un humano experto en el área del problema?

2) ¿Se pretende obtener en un programa de computadora versatilidad o especialización en las diferentes áreas del conocimiento humano?

3) Si se intenta llegar a una solución de manera similar a como lo haría un humano, se debe entonces visualizar varias soluciones y de ellas escoger la más viable.

4) Algunos han considerado que la simulación del pensamiento humano es un fracaso si se considera que la computadora no toma en cuenta factores emocionales como son el aburrimiento, la motivación, el interés, la fatiga o la comprensión. Esto repercute en que las soluciones del hombre sean múltiples, interactivas y flexibles, mientras que las de la computadora sean sencillas y fijas.

Conceptos

Cada uno de los puntos anteriores son detallados cuando se estudia la teoría de los Sistemas Expertos. Comenzaremos viendo algunos conceptos que son importantes:

Inteligencia Artificial, es una ciencia encargada de estudiar y diseñar procesos inteligentes por medio de la computadora. Algunos de los campos de la Inteligencia Artificial son los siguientes:

a) Procesamiento del Lenguaje Natural: trata sobre la comprensión inteligente por parte de la máquina a las formas de expresión del hombre, esto permite una mejor interacción entre el hombre y la computadora.

b) Sistemas Reconocedores de Imágenes: son aquellos que son capaces de interpretar el significado de imágenes.

c) Robótica: es el estudio y diseño de dispositivos electromecánicos (robots) con la capacidad de realizar acciones inteligentes.

d) Sistemas Expertos: es el estudio y desarrollo de conjuntos de programas de computadora con la capacidad de comportarse como un experto humano ante ciertos problemas dentro de un área del conocimiento.

Definición

Un sistema experto es un programa o conjunto de programas que emplea las técnicas de la Inteligencia Artificial para la solución de problemas. La solución que éste dará será una solución dada al nivel de un experto humano. Un Sistema Experto busca una solución satisfactoria tal que sea lo suficientemente exacta de acuerdo al dominio del problema, aunque no sea la óptima.

El Sistema Experto, lejos de pretender versatilidad, busca generar soluciones de una gama de problemas dentro de una rama del conocimiento. Se ha logrado simular procesos tales como la experiencia, de manera que las soluciones de los Sistemas Expertos son aplicables en general.

Sistemas Expertos frente a Sistemas Tradicionales

La principal diferencia entre un programa tradicional y un Sistema Experto, radica en la manera en la que se resuelve el problema. Un sistema tradicional, está hecho para la solución de un problema específico, de manera que si cambia un poco la estructura del problema, deberá ser modificado el programa. Podemos decir que en un sistema tradicional, ante el planteamiento del problema, el proceso de razonamiento lo lleva a cabo el analista, logrando un programa más sencillo. La principal ventaja de estos sistemas es la reducción de tiempo para la generación de los programas que son más sencillos. La desventaja es la limitación para solucionar problemas más complejos, teniéndose que modificar los programas cuando cambia un poco la estructura del problema.

Un Sistema Experto está diseñado para generar soluciones de una gama de problemas dentro de un dominio determinado. Aunque la estructura de los problemas cambiase, el Sistema está capacitado para dar soluciones y en el caso de requerir de mayor información, se puede añadir conocimiento sin necesidad de modificar el método de solución. Aquí el proceso de razonamiento lo lleva propiamente el Sistema Experto. Entre las ventajas de utilizar un Sistema Experto sobre un sistema tradicional, se encuentran las siguientes:

- Se tiene la posibilidad de enfrentar una cantidad más grande de problemas que si se utilizan métodos de solución directa (sistemas tradicionales).
- Dada la capacidad de razonamiento del sistema, se puede dar solución a problemas muy complejos, incluso a aquellos que son difíciles para el hombre.

- Debido a que el mecanismo de razonamiento se separa del conocimiento, éste se puede modificar sin necesidad de afectar al primero.

Pongamos como ejemplo un problema y veamos las dos maneras de resolverlo:

Problema de las Torres de Hannoi: Se tienen tres agujas y en la primera se tienen 8 discos (el problema original habla de 64 discos) apilados, cada uno ligeramente menor al que está debajo de él, como se muestra en la figura 1.1.

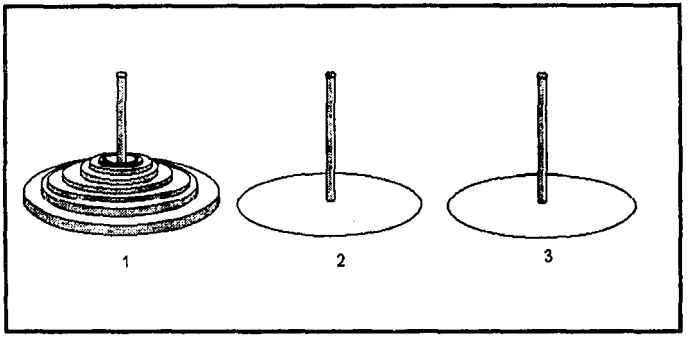

Figura 1.1. El problema de las Torres de Hannoi

La tarea consiste en pasar todos los discos de la aguja 1 a la aguja 3 con dos condiciones: sólo puede moverse un disco a la vez y ningún disco podrá ponerse sobre otro más pequeño.

El método tradicional requeriría que el programador razonara los movimientos que habría que hacerse, obtener el algoritmo, codificarlo y realizar la tarea.

La aplicación del Sistema Experto contemplaría, por ejemplo, la acción mover con los siguientes parámetros:

mover(d, a1, a2)

en donde se moverá el disco número d, de la aguja a1 a la aguja a2, Tomaría en cuenta también los estados de cada aguja:

a1(i)
a2(j)
a3(k)

en donde a1, a2 y a3 son las agujas y los números i,j,k la cantidad de discos que contienen cada una respectivamente.
Tomando en cuenta la acción mover y los estados de las agujas, el Sistema Experto generaría su propio método de solución para llegar al estado en que la aguja a3 contiene todos los discos apilados cada uno con uno ligeramente mayor debajo de él (los estados serían a1(0), a2(0), a3(8)).

En conclusión, diremos que mientras en un sistema tradicional, el proceso de razonamiento lo lleva a cabo el analista, en un Sistema Experto lo lleva a cabo el programa y como dicen algunos autores, se sigue aquella filosofía de que a la larga es mejor enseñar a pescar al hambriento que darle un pez.

APLICACIONES DE LOS SISTEMAS EXPERTOS

Es preciso decir que ante situaciones determinadas no es óptimo usar un Sistema Experto, debido principalmente a razones de tiempo y esfuerzo. La aplicación de Sistemas Expertos será adecuada en casos como los siguientes:

1) Situaciones en las que existen expertos humanos con conocimientos complejos y perfectamente delimitados y en donde los problemas no tienen solución mediante un algoritmo predeterminado, sino que se requiere de ciertos razonamientos para llegar a la solución.

2) Situaciones en donde existen teorías desarrolladas pero en donde es imposible el análisis de todas las posibles rutas de solución.

En ambos casos, el experto llega a la solución gracias a sus conocimientos y experiencias. El Sistema Experto es entonces un programa capaz de aplicar conocimientos y experiencia para llegar a la solución de un problema.

La implementación de un Sistema Experto permite por un lado, evitar fallas en tareas rutinarias y obtener diagnósticos confiables con gran rapidez y por otro lado, ocasiona que el experto pueda desarrollarse más en otros campos quitando su atención de trabajos rutinarios.

I. La Teoría de los Sistemas Expertos

Los Sistemas Expertos han encontrado aplicación en diferentes sectores como son: Bancos, Industria, Comercios, Instituciones de Servicios, Empresas Estatales, Hospitales, etc.

En general los problemas se resuelven de dos formas: análisis mediante razonamiento inductivo o síntesis mediante razonamiento deductivo.

Análisis mediante razonamiento inductivo

Con el análisis, se parte de un problema complejo para dividirlo en otros más pequeños y resolverlo. El razonamiento inductivo va de lo particular a lo general, es decir, en base a ciertos hechos y premisas específicas se intenta alcanzar una conclusión general. Algunos ejemplos de razonamiento por inducción son:

- Diagnóstico: identificar problemas.
- Predicción: se pronostican resultados a partir de cierta información.
- Pruebas: se buscan hechos que se ajusten a determinados criterios.
- Clasificación: se agrupan distintos elementos en categorías distintas.

Cada una de estas tareas puede ser realizada por un sistema experto.

Síntesis por razonamiento deductivo

Sintetizar significa trabajar con hechos y alcanzar una solución o meta. El razonamiento deductivo emplea premisas muy generales para llegar a una conclusión, es decir, va de lo general a lo específico. Algunos problemas que requieren este método son:

- Diseño: crear algo nuevo y original a partir de ciertos conocimientos.
- Planificación: desarrollar una secuencia de pasos para alcanzar un objetivo.
- Configuración: seleccionar y montar partes para formar un todo.

Estas tareas también pueden resolverse con un sistema experto.

Las aplicaciones para las que se emplean los sistemas expertos son las siguientes:

a) *Análisis e interpretación:* se utilizan sistemas expertos para analizar grandes cantidades de información y proporcionar alguna recomendación. En este sentido los sistemas expertos son herramientas muy útiles si consideramos que es difícil y tardado para una persona manejar grandes cantidades de información y hacer conclusiones. Ejemplos de estas aplicaciones son los sistemas de análisis en

planificación financiera, los cuales reciben como entrada todos los datos financieros de un cliente junto con sus propósitos. El sistema analiza todas las oportunidades de inversión y proporciona un plan de acción adecuado.

b) *Predicción:* los sistemas expertos son utilizados para predecir resultados o deducir consecuencias futuras en base a datos y hechos. La base de conocimientos del sistema experto contiene patrones para predecir de manera fiable. Sistemas de predicción son empleados en el área meteorológica y en la inversión de bolsa. Existe un sistema experto muy famoso llamado *Prospector,* el cual es utilizado para encontrar depósitos de mineral y que ha dado muy buenos resultados.

c) *Diagnóstico y depuración:* los sistemas expertos pueden utilizarse para detectar errores y sugerir acciones correctivas. Esto es muy empleado en la reparación y detección de fallas en equipos y también en medicina de diagnóstico. El sistema experto primero recopila los datos necesarios sobre el sistema u organismo y enseguida relaciona todo el conocimiento existente en él para dar un diagnóstico. Ejemplos de este tipo de sistemas son ACE, empleado para detectar errores en redes telefónicas y PUFF, empleado en medicina para la detección de enfermedades pulmonares.

d) *Control:* se utilizan sistemas expertos también para el control de procesos. Generalmente, una computadora recibe valores de variables mediante sensores, dichos valores son verificados por el sistema experto. El sistema toma acciones para corregir los valores de las variables a aquellos que se desea tener en base a los criterios de su base de conocimientos. Existen sistemas de control llamados *de lazo cerrado,* en los cuales se utiliza el concepto de retroalimentación, es decir, se mide el valor real de las variables y se compara con el que se desea tener, entonces se toman las acciones correspondientes.

e) *Diseño:* se refiere a la creación de un nuevo producto, dispositivo o procedimiento. En esta área son utilizados los sistemas expertos dado que el diseño implica la aplicación de muchos conocimientos. Algunos ejemplos son PEACE y SADD empleados para el diseño de circuitos electrónicos y algunos programas de CAD[1] (diseño asistido por computadora) para la creación de autos.

f) *Planificación:* elaborar un plan significa constituir una serie de acciones para alcanzar una meta. Cuando las personas, los recursos, las acciones y las previsiones involucradas crecen, resulta difícil establecer un plan sin ayuda de una computadora. Actualmente se emplean sistemas expertos en planes para la productividad, logística, fabricación y gestión de inventarios, etc.

g) *Enseñanza:* un sistema experto puede utilizarse para la enseñanza dado el conocimiento que almacena. Actualmente existen sistemas de enseñanza asistida por computadora que muestran deficiencias al tratar de adaptarse al ritmo de

[1] CAD, Computer-Aided Design

aprendizaje de cada persona. Esto es mejorado por los sistemas expertos, los cuales si bien es cierto no se han desarrollado mucho en el área educativa, aquellos que existen han dado resultados satisfactorios.

Muchas veces se tiene la idea de que la aplicación de Sistemas Expertos no es real, sin embargo no es así. En áreas como la Industria se tienen aplicaciones para procesos como los siguientes: control y gobierno de procesos, supervisión de estados, diseño de productos e instalaciones, sistemas para el diagnóstico de fallas, sistemas para la planificación de proyectos, sistemas para el asesoramiento de clientes de acuerdo a sus necesidades, sistemas para la formación y adiestramiento de recursos humanos. La aplicación de Sistemas Expertos en estos casos permite ahorrar tiempo y recursos.

Por otro lado, aunque los sistemas expertos intenten tener un razonamiento similar al de los humanos, el propósito no es remplazar a los mismos. Los sistemas expertos son una herramienta para realizar el trabajo de una manera rápida, fácil y completa, proporcionando información para enfrentar mejor un problema. Sin embargo el sistema experto no va a tomar decisiones, la decisión final la tomará la persona, la cual podrá seguir o no la recomendación del sistema de acuerdo a su juicio.

ELEMENTOS DE UN SISTEMA EXPERTO

Los diferentes autores señalan varias arquitecturas para los Sistemas Expertos, sin embargo la única diferencia entre ellas es sólo el nombre de los componentes. La característica principal en la arquitectura de un Sistema Experto es el hecho de que el conocimiento esté separado del mecanismo de generación de soluciones, lo cual permite en principio, poder distinguir ya dos elementos. Podemos decir que cualquier Sistema Experto debe contener las siguientes partes:

1) Base de conocimientos: es el lugar en donde reside toda la información disponible para la solución de problemas. Se dice que en buena parte el éxito de un Sistema Experto está en proporción con los conocimientos que maneje.

2) Motor de Inferencia: es la parte del Sistema que se encarga de hacer las búsquedas necesarias en la base de conocimiento, para generar la solución. El motor de inferencia hará uso de los conocimientos y heurísticos necesarios para obtener una o varias soluciones.

3) Módulo de comunicación con el usuario: se encarga de lograr la interacción entre el usuario y el sistema experto; entre sus funciones están el recibir datos y emitir resultados, soluciones y explicaciones.

4) Memoria de trabajo: es una representación en donde existen afirmaciones de aplicación específica. Dichas afirmaciones están compuestas tanto por los datos que son proporcionados por el usuario como por los resultados parciales que el sistema experto genera.

5) Módulo de comunicación con el experto: es aquel que permite la interacción del sistema experto con el ingeniero de conocimiento, es utilizado para integrar más conocimiento al sistema.

En los subcapítulos posteriores se describe con más detalle cada parte de un sistema experto.

I.2

La Representación del Conocimiento

LA BASE DE CONOCIMIENTO

Como hemos mencionado antes, la base de conocimiento es aquella parte del sistema experto en donde reside el conocimiento en un formato determinado. El ingeniero de conocimiento es la persona encargada de obtener el conocimiento necesario para el funcionamiento del sistema experto y darle el tratamiento necesario para poder ser procesado por la computadora. Para lograr su objetivo, el ingeniero suele:

a) Leer literatura relacionada con el dominio del sistema para familiarizarse con el tema y la terminología.

b) Sostener una o varias entrevistas con uno o varios expertos humanos para adquirir su conocimiento.

c) Organizar los resultados obtenidos de estas entrevistas y escribirlos en forma de software que una computadora pueda utilizar.

La base de conocimiento contiene tres tipos de componentes del conocimiento:

a) Hechos: son declaraciones dentro del dominio del sistema, que siempre se cumplen, por ejemplo:
- Una manzana es roja.
- Un aspirante a médico tiene espíritu de servicio.
- Un Boeing 747 vuela sin problemas con tres motores.

b) Reglas de procedimiento: son secuencias invariables que relacionan ciertos eventos dentro del dominio del sistema, por ejemplo:

- Siempre verifique el movimiento vehicular antes de entrar a una autopista.
- Antes de abrir un televisor, asegúrese de que esté desconectado.

c) Reglas heurísticas: es conocimiento adquirido de manera empírica que puede aplicarse cuando no existen reglas de procedimiento aplicables. Las reglas heurísticas son producto de la experiencia de los expertos. Por ejemplo:

- Es mejor intentar un aterrizaje de emergencia bajo condiciones controladas que volar en condiciones desconocidas.

Representación del conocimiento

Una vez recopilado el conocimiento, debe ser puesto en una forma reconocible por la computadora. Actualmente existen varias estructuras establecidas para la representación del conocimiento de las cuales deberá elegirse la más adecuada al dominio del sistema experto. La estructura de la base de conocimiento afecta de manera directa el funcionamiento del sistema experto, por lo que al elegirla deben tomarse en cuenta aspectos como los siguientes:

- Transparencia: la estructura debe permitir que el conocimiento almacenado pueda ser interpretado fácilmente.

- Naturalidad: al representar el conocimiento se debe procurar que este no pierda sus características natas.

- Eficiencia: la estructura debe dar facilidades al motor de inferencia para utilizar el conocimiento representado.

- Adecuación: la estructura debe ser capaz de almacenar todo el conocimiento requerido por el Sistema Experto.

- Modularidad: que el conocimiento pueda almacenarse en fragmentos de manera independiente.

Un sistema de representación del conocimiento consta de cuatro partes fundamentales:

- Una parte de léxico, que determina que símbolos está permitido usar en la representación.
- Una parte estructural, que establece la forma en la que se ordenan los símbolos.
- Una parte operativa, que determina los procedimientos de acceso al conocimiento para poder utilizarlo.
- Una parte semántica, que establece la forma de interpretar la representación del conocimiento.

A continuación se describen las siguientes estructuras para la representación del conocimiento:

a) Lógica de predicados
b) Reglas de producción
c) Plantillas

LOGICA DE PREDICADOS

De la manera más general, existen dos tipos de representación de conocimiento:

a) Representación formal del conocimiento
b) Representación no formal del conocimiento

Dentro de la representación formal encontramos la lógica formal, la cual es resultado de antiguas consideraciones filosóficas y el sistema más usado de lógica formal es la Lógica de Predicados.

La lógica de predicados está compuesta de varios elementos:

1. Un alfabeto
2. Un lenguaje formal
3. Un conjunto de enunciados básicos llamados axiomas
4. Un conjunto de reglas de inferencia

El alfabeto es el conjunto de símbolos a partir de los cuales se construirán los enunciados, consta de:

a) Constantes: son usadas para especificar un elemento determinado del dominio. Se representan por letras mayúsculas. Por ejemplo:

MANZANA una fruta

PSICOLOGIA una carrera

b) Variables: representan un conjunto de elementos del dominio sin especificar un elemento determinado. Se representan con letras minúsculas. Por ejemplo:

fruta cualquier fruta

carrera cualquier carrera

c) Predicados: se emplean para representar relaciones entre los elementos del dominio. Un predicado junto con los elementos que relaciona, forma una fórmula atómica o átomo, el cual es el elemento básico para la lógica de predicados. Por ejemplo:

CITRICO(LIMON) Limón es un cítrico

MASALTO(TITO,PEPE) Tito es más alto que Pepe

Para la construcción de proposiciones compuestas es común utilizar algunas de las siguientes conjunciones:

^ and (y)
v or (O) inclusivo
→ implica
≡ equivalente a
~ no (negación)

Los cuales tienen el siguiente significado:

^ Utilizada para construir fórmulas en donde los componentes deben ser verdaderos para que la fórmula sea verdadera. Por ejemplo, para expresar que mi auto es azul, compacto y automático,

AZUL(MI-AUTO)∧COMPACTO(MI-AUTO)∧AUTOMATICO(MI-AUTO)

∨ Se usa para construir fórmulas cuya veracidad depende del cumplimiento de alguno de los elementos. Por ejemplo, para expresar que Juan compró un departamento o una casa,

COMPRO(JUAN,DEPARTAMENTO)∨COMPRO(JUAN,CASA)

~ Se emplea para cambiar el valor de una fórmula de verdadero a falso o de falso a verdadero. Por ejemplo, para representar la frase Laura no es un muchacho:

~MUCHACHO(LAURA)

→ Utilizado para realizar construcciones del tipo Si-Entonces (if-then). El cumplimiento del antecedente implica que el consecuente sea verdadero también. Por ejemplo:

[APELLIDO(PEREZ)∧PILA(JUAN)]→NOMBRE(JUAN PEREZ)

≡ Utilizado para expresar la equivalencia de dos fórmulas. Por ejemplo, x≡y implica que la veracidad del lado derecho y del lado izquierdo son equivalentes.

Cuantificación

Cuando es necesario expresar la validez de una expresión para cierto rango de elementos, se usan los cuantificadores. El cuantificador universal ∀x indica que la fórmula es verdadera para todos los valores de la variable implicada. Por ejemplo, para indicar que toda persona necesita aire:

∀x[PERSONA(x)→NECESITA-AIRE(x)]

El cuantificador existencial ∃x indica que la fórmula se cumple al menos para un valor de la variable. Por ejemplo:

∃x[PROPIETARIO(x,AUTOMOVIL)∧PROPIETARIO(x,BOTE)]

La fórmula se cumplirá para aquella persona que sea dueña del auto y del bote.

Lenguaje

El lenguaje de una lógica de predicados es el conjunto de todas las posibles fórmulas que pueden hacerse dado el alfabeto de símbolos. Una fórmula válida para el lenguaje se llama una fórmula bien definida (**FBD**) y se define inductivamente como sigue:

1. Una fórmula atómica es una FBD.

2. Las siguientes son FBD:
   (~F),(F∧G),(F∨G),(F→G)
   donde:
   F y G son FBD
3. Las siguientes son FBD:
   (∀xF) y (∃xF)
   donde:
   F es una FBD y x una variable.

Evaluación de la Lógica de predicados

La evaluación de las FBD de una lógica de predicados se acercan al ideal de un sistema experto, ya que podemos decir que cualquier FBD puede ser sometida a un método riguroso para comprobar su veracidad.

Podemos dividir la evaluación de una lógica de predicados en dos partes:

1) La básica, la evaluación de una fórmula atómica: el valor de verdad de un átomo se logra mediante la interpretación del predicado. Por ejemplo, para encontrar el valor de verdad del átomo ADULTO(JOSE), tendríamos que conocer que el predicado ADULTO(x) será cierto cuando x tenga una edad mayor o igual a 18 años, de esta manera, para conocer el valor de ADULTO(JOSE) tendríamos que leer la edad de JOSE y entonces determinar si el átomo es verdadero o falso.

2) Evaluación de fórmulas complejas: en este caso se utiliza un método llamado *método de la tabla de verdad*, el cual se basa en la siguiente tabla de verdad:

X	Y	X∧Y	X∨Y	X→Y	~Y	X≡Y
V	V	V	V	V	F	V
V	F	F	V	F	F	F
F	F	F	F	V	V	V
F	V	F	V	V	V	V

Tabla 1.2 *Tabla de verdad para la evaluación de fórmulas complejas.*

Para determinar el valor de una fórmula compleja, se procede a evaluar cada átomo desde adentro hacia afuera auxiliándose de la anterior tabla de verdad. Por ejemplo, evaluemos la siguiente fórmula suponiendo que el predicado P es falso y que los predicados Q y R son verdaderos:

$$[(P \vee (Q \wedge R)) \wedge \sim P] \rightarrow (Q \wedge P)$$

Procediendo a evaluar, obtenemos lo siguiente:

$$[(P \vee V) \wedge \sim P] \rightarrow (Q \wedge P)$$

$$[F \vee \sim P] \rightarrow (Q \wedge P)$$

$$[F \vee V] \rightarrow (Q \wedge P)$$

$$V \rightarrow F$$

$$F$$

Obteniendo como resultado final que la expresión es falsa.

Ingeniería de conocimiento para lógica de predicados

Para representar el conocimiento bajo la lógica de predicados, se recomienda seguir los siguientes pasos:

1) Tener una comprensión del conocimiento.
2) Representar el conocimiento mediante enunciados en español.
3) Identificar los componentes de los enunciados.
4) Escoger símbolos para representar cada componente de los enunciados.
5) Construir los FBD que representen los enunciados, utilizando los símbolos definidos anteriormente.

Ejemplo 1: para representar la siguiente expresión en fórmula de predicados: "Pedro vive en Morelia pero no en Parral o Zamora", primero entendemos que Pedro vive en Morelia pero no vive en Parral ni en Zamora, los enunciados serían los siguientes:

1) Pedro vive en Morelia.
2) Pedro no vive en Parral.
3) Pedro no vive en Zamora.

Formulando nuestra FBD:

VIVE_EN (PEDRO, MORELIA) $\wedge\sim$ VIVE_EN (PEDRO, PARRAL) $\wedge\sim$ VIVE_EN (PEDRO, ZAMORA)

Ejemplo 2: representar en lógica de predicados la siguiente expresión: "si usted no hace ejercicio, usted aumentará de peso". En este caso debemos emplear el cuantificador universal ya que la expresión se generaliza para cualquier persona:

$$\forall x [\sim \text{EJERCICIO} (X) \rightarrow \text{AUMENTO\_PESO} (X)]$$

Procesos de inferencia para la lógica de predicados

Un proceso de inferencia tiene como objetivo el generar nuevas FBD a partir de las FBD existentes. En la lógica de predicados existen dos formas de realizar inferencia:

1) Mediante la aplicación de las siguientes reglas de inferencia:

(a) Regla *modus ponens:*

$$[P1 \wedge (P1 \rightarrow P2)] \rightarrow P2$$

lo cual indica "si P1 es verdadera y P1 siendo cierta implica que P2 es verdadera, entonces P2 es verdadera". Por ejemplo, si la presencia de risa implica un estado de alegría, entonces mediante la aplicación de *modus ponens* podemos inferir que hay alegría al observar risa.

(b) Regla de *especialización universal*

$$\text{INDIVIDUO} \wedge (\forall x)\,[\,F\,(X\,)\,] \rightarrow F\,(\,\text{INDIVIDUO}\,)$$

En esta regla se genera una FBD para un elemento específico a partir de una FBD generalizada. Por ejemplo, aplicando esta regla podríamos deducir que Pedro respira de la FBD que nos dice que todas las personas respiran.

2) El *método de sustitución:* se basa en la aplicación de ciertas leyes que son válidas para cualquier FBD, dichas leyes se muestran en la siguiente tabla:

Leyes de equivalencia para lógica de predicados

(1)	$P1 \rightarrow P2 \equiv \sim P1 \vee P2$	
(2)	$P1 \vee P2 \equiv P2 \vee P1$	(conmutativa)
(3)	$P2 \wedge P1 \equiv P1 \wedge P2$	(conmutativa)
(4)	$P1 \vee (P2 \wedge P3) \equiv (P1 \vee P2) \wedge (P1 \vee P3)$	(distributiva)
(5)	$P1 \wedge (P2 \vee P3) \equiv (P1 \wedge P2) \vee (P1 \wedge P3)$	(distributiva)
(6)	$(P1 \vee P2) \vee P3 \equiv P1 \vee (P2 \vee P3)$	(asociativa)
(7)	$(P1 \wedge P2) \wedge P3 \equiv P1 \wedge (P2 \wedge P3)$	(asociativa)
(8)	$P1 \rightarrow P2 \equiv \sim P1 \rightarrow \sim P2$	
(9)	$\sim(\sim P1) \equiv P1$	
(10)	$\sim(P1 \vee P2) \equiv \sim P1 \wedge \sim P2$	(de D'Morgan)
(11)	$\sim(P1 \wedge P2) \equiv \sim P1 \vee \sim P2$	(de D'Morgan)
(12)	$P1 \vee \text{FALSO} \equiv P1$	
(13)	$P1 \vee \text{VERDADERO} \equiv P1$	
(14)	$P1 \wedge \text{VERDADERO} \equiv P1$	
(15)	$P1 \wedge \text{FALSO} \equiv \text{FALSO}$	
(16)	$P1 \vee \sim P1 \equiv \text{VERDADERO}$	
(17)	$P1 \wedge \sim P1 \equiv \text{FALSO}$	

Tabla 1.3

Ejemplo: Dados los siguientes hechos:

1. Cuando un avión se queda sin combustible, los motores se apagan.
2. Si los motores de un avión se detienen, se requiere un aterrizaje forzoso.
3. En un momento dado, el avión del capitán J. A. Sánchez se queda sin combustible.

demostrar que los motores del avión se detienen y que se requiere de un aterrizaje forzoso cuando se agota el combustible.

Para construir un FBD a partir de los hechos dados, debemos introducir primero los siguientes predicados:

SC El avión del capitán Sánchez se quedó sin combustible
MD Los motores del avión del capitán Sánchez se detienen
AF Se requiere un aterrizaje forzoso del avión del capitán Sánchez

Construyendo la FBD:

$$(SC \rightarrow MD) \wedge (MD \rightarrow AF) \wedge SC$$

por 1:
$(\sim SC \vee MD) \wedge (\sim MD \vee AF) \wedge SC$

por 3:
$SC \wedge (\sim SC \vee MD) \wedge (\sim MD \vee AF)$

por 5:
$(SC \wedge \sim SC) \vee (SC \wedge MD) \wedge (\sim MD \vee AF)$

por 17;
$FALSO \vee (SC \wedge MD) \wedge (\sim MD \vee AF)$

por 12:
$SC \wedge MD \wedge (\sim MD \vee AF)$

por 7:
$SC \wedge [MD \wedge (\sim MD \vee AF)]$

por 5:
$SC \wedge [(MD \wedge \sim MD) \vee (MD \wedge AF)]$

por 17:
$SC \wedge [FALSO \vee (MD \wedge AF)]$

por 12:
SC ∧ MD ∧ AF

y finalmente con la obtención de está FBD comprobamos lo que deseábamos.

Conclusiones

Como mencionamos anteriormente, la principal ventaja de la lógica de predicados es que su método de inferencia permite evaluar la veracidad de cualquier FBD. Sin embargo, como pudimos ver en el ejemplo anterior, el proceso de inferencia suele ser largo. Al implementar una lógica de predicados, el motor de inferencia debe optimizarse de manera que no se procesen FBD que no sean necesarias, ya que de lo contrario estaríamos gastando gran cantidad de tiempo de manera innecesaria.

SISTEMAS DE PRODUCCION

Los sistemas de producción son los más comúnmente usados en los sistemas expertos, un sistema de producción basa su representación de conocimiento en reglas de producción. Dentro de los sistemas de producción identificamos las siguientes partes:

1. Un área de memoria que contiene el estado actual de los elementos.
2. Un conjunto de reglas de producción.
3. Un analizador encargado de emplear las reglas adecuadas.

Memoria global

En la memoria global se almacenan los valores actuales de los elementos en la forma *identificador del elemento* y en seguida una seria de pares *atributo-valor.* Por ejemplo:

Ana | ∧color_ojos café ∧color_cabello castaño ∧estatura 1.70 ∧ estado_ánimo alegre ∧edad 22 ∧labor jugando

Reglas de producción

Una regla de producción se constituye de dos partes: una parte condicional y una parte de acciones. La parte condicional, también llamada lado izquierdo, describe las condiciones que deben ocurrir para que la regla pueda ser aplicada (disparada). En esta parte se hace uso de los elementos de la memoria global: identificadores, atributos y valores.

La parte de acción de una regla, también conocida como lado derecho (LD) describe las acciones que se ejecutarán cuando la regla sea disparada. Las acciones más comunes son cambiar los valores de los atributos de la memoria global, generar nuevos estados de los elementos o realizar una actividad específica definida por el usuario.

La sintaxis común de las reglas de producción es la siguiente:

(p < nombre producción >
(< elemento condicional >)
→
(< acción >)
.
.
.

donde:

< nombre producción > es el nombre que identifica la regla de producción.

< elemento condicional > = < identificador del elemento de memoria >
∧ < identificador de atributo > < valor >
.
.
.
∧ < identificador de atributo > < valor >

donde:
< identificador atributo > es el atributo cuyo valor deberá compararse
< valor > es una constante o variable que especifica el valor que debe tener el atributo para que se
cumpla la condición.

el símbolo → significa "implica".

< acción > es la acción que se tomará al cumplirse la condición.

Las acciones más usuales son las siguientes:

hacer: adicionar un nuevo elemento a la memoria global.
remover: eliminar un elemento de la memoria global.
modificar: modificar el valor de un determinado atributo del elemento de memoria indicado.
calcular: calcular un valor a partir de las variables.
leer: recibir valores de parte del usuario.
escribir: dar una salida al usuario.
llamar: ejecutar un procedimiento específico definido por el usuario.

Por ejemplo, para el ejemplo anterior, indiquemos que Ana se pone pensativa cuando estudia:

(p estudia
(Ana ∧labor estudiando) →
(modificar ∧estado_ánimo pensativo))

Representación factual y procedimental

Como mencionamos anteriormente, el conocimiento se compone de hechos (representación factual) y de heurísticos (representación procedimental). Ambos tipos de conocimiento pueden ser representados por reglas de producción. Por ejemplo, para representar el hecho de que "los jitomates son rojos", podríamos construir la siguiente regla de producción:

(p Color jitomate
(VEGETAL ∧clase jitomate) →
(modificar VEGETAL ∧color rojo))

Para representar un enunciado como "si hay goteras cierre bien la llave" podemos emplear una regla como la siguiente:

(p cerrar_llave
(llave ∧ estado gotea) →
(modificar llave ∧ estado cerrar)

Interpretador

Un interpretador sencillo de un sistema de producción, lee el LI de cada regla y de acuerdo con los valores encontrados en memoria global, ejecuta aquella regla que se cumpla. De esta manera, el ciclo del interpretador es:

1) Instanciar cada regla, es decir, verificar si la condición dada en la regla se cumple de acuerdo con el estado actual de los elementos de memoria global.

2) Identificar que regla de producción se ejecutará. En grandes sistemas de producción, existe una cantidad grande de reglas que pueden ser disparadas, de manera que se requiere de un criterio que pueda decirnos que regla será ejecutada. Hay una estrategia muy común, usada para resolver este tipo de conflictos, dicha estrategia dice lo siguiente:

1. Descartar todas las reglas que ya se hayan disparado.

2. Seleccionar la regla cuyos LI referencia los elementos de memoria que fueron más recientemente agregados a la memoria global. De esta manera se logra un encadenamiento de reglas, logrando secuencias más grandes.

3. Seleccionar la instancia que tenga un LD más detallado.

4. Seleccionar una instancia casi en forma arbitraria.

5. Realizar la acción de la instancia seleccionada.

Conclusiones

Actualmente, los sistemas de producción son los más usados en sistemas expertos y se dice que aquellos sistemas expertos que han logrado mayor éxito basan la representación de su conocimiento en reglas de producción. Principalmente, los sistemas de producción tienen aceptación por la forma en la que representan el conocimiento, ya que en muchas ocasiones la gente prefiere expresar su conocimiento en la forma "si - entonces". Los sistemas de producción, debido a la independencia de las reglas, permiten adicionar conocimiento sin tener que hacer cambios en otras partes del sistema. Una desventaja que encontramos en estos sistemas es la falta de un control algorítmico para llevar a cabo una cierta secuencia de reglas.

OBJETOS ESTRUCTURADOS

Otra de las formas de representación del conocimiento es mediante objetos estructurados. El término objetos estructurados engloba las siguientes denominaciones:

- Marcos.
- Esquemas, en trabajos sobre la memoria.
- Grafismos, que describen el encadenamiento estereotipado de sucesos.
- Prototipos, o unidades en KRL.
- Objetos.
- Formas y clases, en Simula.
- Tipos abstractos.
- Plantillas.

En lo subsiguiente emplearemos el término *plantilla*.

Dentro de esta forma de representación del conocimiento, encontramos a la *plantilla* como la entidad básica para la construcción del conocimiento. Una plantilla contiene descriptores (o slots) que son los parámetros que le caracterizan. Estos descriptores tienen valores estándares. A su vez, existe una jerarquía en donde podemos identificar varios objetos que pertenecen a una misma *clase*.

Aquí se manejan los siguientes conceptos:

a) Plantilla: es la entidad lógica base. Una plantilla contiene datos y un código que manipula dichos datos. Dentro de una plantilla puede haber datos o código que sea privado, es decir, no se pueden acceder desde afuera de él, a esto se le conoce como encapsulamiento.

b) Polimorfismo: se refiere a la capacidad de usar un nombre en general, el cual se empleará para diferentes tipos de datos. Por ejemplo, permite la existencia de una función de un mismo nombre que pueda aplicarse a datos de diferente tipo.

c) Herencia: es la transferencia de características de una plantilla a otra. Por ejemplo, el objeto silla hereda las características del objeto mueble y especifica además características propias. La herencia permite hacer uso de un objeto específico dentro de una clase más general.

Veamos el siguiente ejemplo:

SILLA :
Clase de (valor mueble)
Número de patas (defecto 4)
Color ((posibilidades (blanca azul marrón) (defecto marrón)
Edad (restricción (>0) (<700))

SILLA DE MI ABUELA
Clase de (valor silla)
Número de patas (valor 3)
Color (valor blanco)
Material (valor madera)
Edad (valor 50)

donde:

- Valor designa el valor del descriptor.
- Defecto indica el valor del descriptor si no se especifica otra cosa.
- El objeto SILLA DE MI ABUELA tendría por herencia, el valor 4 en su descriptor *número de patas*, sin embargo tiene el valor 3 puesto que es especificado.
- Restricción es una lista de predicados que devolverán CIERTO cuando se aplican al valor al que se trata de afectar el atributo apropiado.
- Posibilidades, son casos particulares de restricción que se indican mediante una lista exhaustiva de los valores posibles.
- Intervalo es otra forma de expresar una restricción. Ejemplo, Edad (intervalo (0 700)).
- Procedimiento, permite llamar a una función que calcula el valor del descriptor.

Razonamiento con plantillas

Para comenzar el proceso de razonamiento, identificamos primero la situación actual y tratamos de seleccionar el objeto que más se parezca. Debido a que en la mayoría de los casos no tendremos un objeto que exactamente se aplique al contexto actual, con frecuencia debemos iniciar con el más adecuado. De esta manera, debemos instanciar los elementos particulares de nuestra situación actual con los elementos particulares de un objeto. Con frecuencia se hace la instanciación con una clase en general y se trata de hallar exactitud en algún objeto de esa clase.

SILLA :
Clase de (valor mueble)
Número de patas (defecto 4)
Color ((posibilidades (blanca azul marrón) (defecto marrón)
Edad (restricción (>0) (<700))

SILLA DE MI ABUELA
Clase de (valor silla)
Número de patas (valor 3)
Color (valor blanco)
Material (valor madera)
Edad (valor 50)

donde:

- Valor designa el valor del descriptor.
- Defecto indica el valor del descriptor si no se especifica otra cosa.
- El objeto SILLA DE MI ABUELA tendría por herencia, el valor 4 en su descriptor *número de patas,* sin embargo tiene el valor 3 puesto que es especificado.
- Restricción es una lista de predicados que devolverán CIERTO cuando se aplican al valor al que se trata de afectar el atributo apropiado.
- Posibilidades, son casos particulares de restricción que se indican mediante una lista exhaustiva de los valores posibles.
- Intervalo es otra forma de expresar una restricción. Ejemplo, Edad (intervalo (0 700)).
- Procedimiento, permite llamar a una función que calcula el valor del descriptor.

Razonamiento con plantillas

Para comenzar el proceso de razonamiento, identificamos primero la situación actual y tratamos de seleccionar el objeto que más se parezca. Debido a que en la mayoría de los casos no tendremos un objeto que exactamente se aplique al contexto actual, con frecuencia debemos iniciar con el más adecuado. De esta manera, debemos instanciar los elementos particulares de nuestra situación actual con los elementos particulares de un objeto. Con frecuencia se hace la instanciación con una clase en general y se trata de hallar exactitud en algún objeto de esa clase.

Como mencionamos anteriormente, una plantilla contiene *descriptores (slots)* que son parámetros que ayudan a la descripción del objeto. También podemos decir que existen ciertos datos llamados *valores de descriptores.*

Existen plantillas que describen una entidad en particular, estas son conocidas como *plantillas de ejemplares o ejemplares.* Hay otras plantillas que describen clases completas, estas se llaman *plantillas de clases o clases.* Las plantillas de ejemplares son instancias de las plantillas de clase.

Existen algunos descriptores especiales, como el descriptor *Es,* que significa "es miembro de la clase" y asigna un ejemplar a la clase de la que es miembro. Por ejemplo, en la *figura 1.4* Alberto es identificado como miembro de la clase Matemáticas.

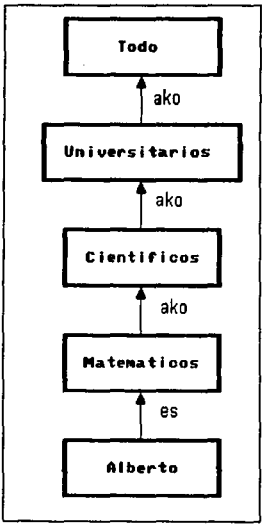

***Figura 1.4** Ejemplo de un sistema de plantillas*

Otro descriptor especial es el descriptor *Ako* (del inglés A kind of) el cual vincula clases entre sí. Por ejemplo, Matemáticas es una *subclase directa* de Científicos ya que existe un solo descriptor Ako entre los dos, y Científicos es una subclase de Todos ya que entre ellos hay dos descriptores Ako. Asimismo, Científicos es una *superclase directa* de Matemáticas y Todos es una superclase de Científicos.

Procedimientos de acceso

Para el acceso a plantillas, existen los siguientes procedimientos:

Constructor de clases: es un procedimiento capaz de crear una plantilla de clase que contenga otros descriptores, ninguna, una o más superclases directas.

Constructor de ejemplares: su función es construir una plantilla de ejemplar. Su entrada es el nombre de la clase a la cual pertenecerá el ejemplar y su salida es un ejemplar de dicha clase. El nuevo ejemplar se conecta de manera automática a su clase directa mediante un descriptor Es.

Escritor de descriptores: fija un valor en un descriptor de una plantilla. Su entrada es una plantilla, el nombre del descriptor y el valor que se va a fijar en él.

Lector de descriptores: recupera los valores de los descriptores. Su entrada es una plantilla y el nombre del descriptor, su salida es el valor que leyó del descriptor.

La herencia y los valores de descriptor

Los valores de los descriptores de un ejemplar están determinados por las superclases de dicho ejemplar. Al crearse un ejemplar, si su superclase tiene un descriptor, entonces el ejemplar hereda tal descriptor. En ocasiones, después de crear un ejemplar, se puede especificar el valor de un descriptor.

Consideremos la siguiente información[2]:

- Los competidores y los golosos de cuentos de hadas son enanos.
- La mayoría de los enanos de cuentos de hadas son gordos.
- El apetito de la mayoría de los enanos de cuentos de hadas es reducido.
- El apetito de la mayoría de los golosos de cuentos de hadas es enorme.
- La mayoría de los competidores de cuentos de hadas son delgados.

[2] Ejemplo extraído del libro *"Inteligencia Artificial", Patrick Henry Winston, Addison-Wesley Iberoamericana.*

Tomemos en cuenta también la jerarquía mostrada en la *figura 1.5.*

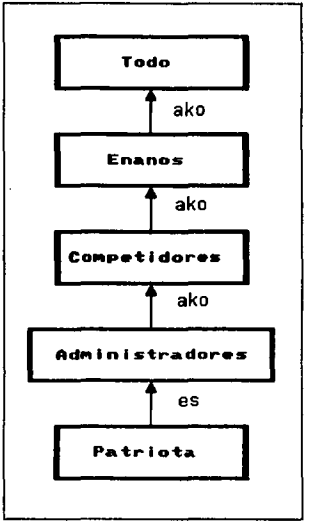

Figura 1.5. Ejemplo de una jerarquía

Podríamos fijar que los Enanos son gordos a menos que se tenga otra información. Al usar la herencia se tienen las siguientes ventajas en el conocimiento:

- es más fácil de construir,
- es más fácil de corregir al cometer un error
- más fácil de actualizar

Una manera de utilizar la herencia, es mediante los procedimientos *cuando-se-construye,* los cuales se asocian a las clases de los ejemplares. Por ejemplo, un procedimiento que asigna un valor para el descriptor *físico* de los Enanos:

Para llenar la ranura Físico cuando se construye un nuevo Enano,

- *escriba Gordo en la ranura.*

Los valores de descriptor establecidos por procedimientos cuando-se-construye son llamados *omisiones.* Cada procedimiento asigna solo una omisión de descriptor para una clase específica, sin embargo, puede haber varios procedimientos cuando-se-construye para clases diferentes, que afecten un descriptor.

Consideremos el siguiente procedimiento cuando-se-construye que proporciona un valor por omisión para el descriptor Físico de la clase Competidores:

Para llenar el descriptor Físico cuando se construye un nuevo Competidor,

- *escriba Delgado en la ranura*

Supongamos que se creará el ejemplar Patriota, el cual es miembro de la clase Administradores. Mientras que el procedimiento asociado a Enanos dice que el Físico es Gordo, el procedimiento asignado a Competidor dice que el Físico es Delgado. Para saber cual procedimiento afectará a Patriota, saquemos la siguiente *lista de precedencia de clases* de la jerarquía de la *figura 2:*

Patriota
Clase Administradores
Clase Competidores (procedimiento asociado)
Clase Enanos (procedimiento asociado)
Clase Todo

Para resolver dicha ambigüedad, se toma el procedimiento que está asociado a la clase más específica del ejemplar, es decir, a la que está primero en la lista de precedencia de clases, en este caso es Competidores; de esta forma queda determinado que el Físico de Patriota es Delgado.

Jerarquías de clase ramificadas

Cuando un ejemplar tiene más de una clase directa o existe más de una superclase directa sobre una clase, se dice que la jerarquía de clase se ramifica. Cuando se tiene una jerarquía ramificada y se tienen varios procedimientos cuando-se-construye que afectan a un descriptor se deben tomar ciertos criterios. Un método para decidir que procedimiento aplicar es una especie de *búsqueda en profundidad exhaustiva de izquierda a derecha* con una condición *subir hasta la unión,* el procedimiento indica lo siguiente para construir la lista de precedencia de clases:

Hacer un recorrido en profundidad de cada rama hasta llegar a un nodo ya visitado. Una vez terminada la rama, recorrer la rama de la derecha, de modo que el recorrido de ramas se hace de izquierda a derecha. La condición subir-hasta-la-unión indica que cualquier clase que se encuentre más de una vez durante toda la búsqueda sea ignorada hasta que se le encuentre por última vez.

Ejemplo: consideremos la jerarquía de clase de la *figura 1.6:*

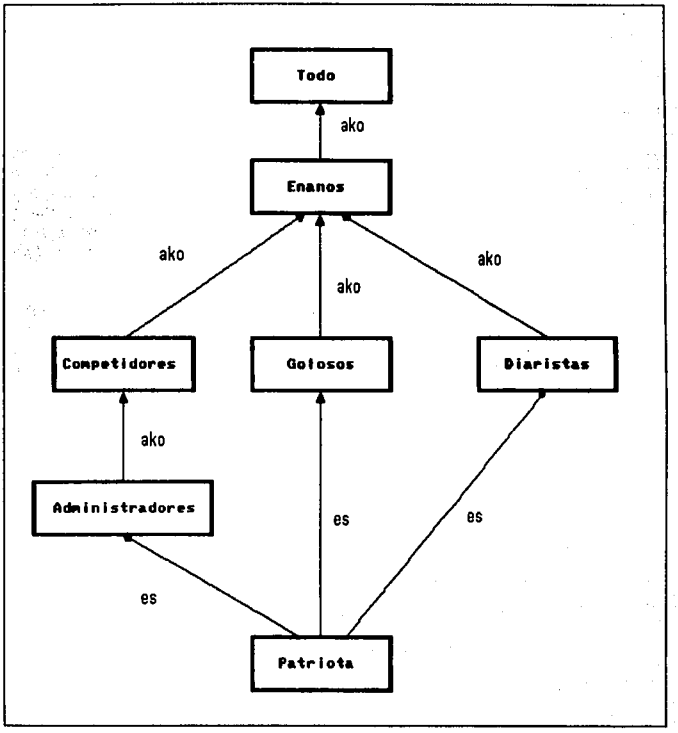

Figura 1.6. *Ejemplo de una jerarquía de clase ramificada.*

sean los dos procedimientos siguientes para calcular el apetito:

Para llenar el descriptor Apetito cuando se construye un nuevo Enano,

- escriba reducido en el descriptor.

Para llenar el descriptor Apetito cuando se construye un nuevo Goloso,

- escriba Enorme en el descriptor.

De acuerdo con el método descrito arriba, la lista de precedencia de clase comienza con el recorrido de la rama extrema izquierda sin tocar Enanos ni Todo ya que se repiten en otras trayectorias, de la rama de en medio solo escribimos Golosos y en el último recorrido incluimos ya Diaristas Enanos y Todo.

Patriota
Administradores
Competidores
Golosos (Procedimiento asociado)
Diaristas
Enanos (Procedimiento asociado)
Todo

De esta manera, como la clase más especifica afecta al ejemplar, deducimos que Patriota tiene Apetito Enorme.

Procedimiento de ordenamiento topológico

Para jerarquías de clase más complicadas, el procedimiento descrito aquí no será suficiente, entonces es necesario utilizar un método llamado *procedimiento de ordenamiento topológico.* El procedimiento se basa en hacer pares de clases adyacentes que permitan construir la jerarquía conservando el orden abajp-arriba e izquierda-derecha. Hagamos un ejemplo con la jerarquía de clase de la figura 2.5. Para establecer los pares adyacentes, hagamos como que ensartamos un anzuelo en el elemento y su superclase directa como se ve en la *figura 1.7:*

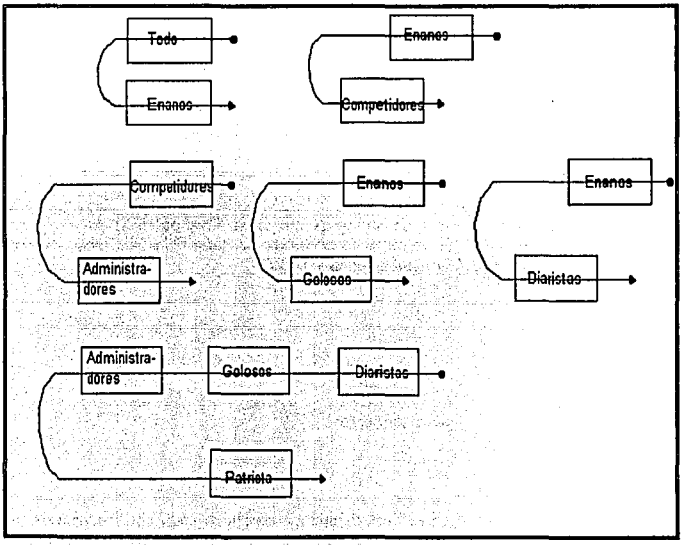

Figura 1.7. *Procedimiento para el ordenamiento topológico*

Ahora establezcamos los pares recorriendo el anzuelo desde la punta y hasta el origen para cada nodo:

Nodo	Pares de anzuelo
Patriota	Patriota-Administradores, Administradores-Golosos, Golosos-Diaristas
Administradores	Administradores-Competidores
Golosos	Golosos-Enanos
Diaristas	Diaristas-Enanos
Competidores	Competidores-Enanos
Enanos	Enanos-Todo
Todo	Todo

El siguiente paso es buscar un elemento en la parte izquierda de los pares que no ocupe la parta derecha en ningún par, a un elemento en estas condiciones le llamaremos expuesto. Para el ejemplo, el elemento expuesto es Patriota. Al hallar un elemento expuesto, agréguelo al final de la lista de precedencia de clases y tache todos los pares en donde esté. Así, tenemos lo siguiente:

Lista de precedencia de clases:

Patriota

Nodo	Pares de anzuelo
Patriota	~~Patriota-Administradores~~, Administradores-Golosos, Golosos-Diaristas
Administradores	Administradores-Competidores
Golosos	Golosos-Enanos
Diaristas	Diaristas-Enanos
Competidores	Competidores-Enanos
Enanos	Enanos-Todo
Todo	Todo

El elemento que ahora queda expuesto es Administradores, por lo que habrá que agregarlo a la lista y tachar sus pares:

Lista de precedencia de clases:

Patriota
Administradores

Nodo	Pares de anzuelo
Patriota	~~Patriota-Administradores~~, ~~Administradores-Golosos~~, Golosos-Diaristas
Administradores	~~Administradores-Competidores~~
Golosos	Golosos-Enanos
Diaristas	Diaristas-Enanos
Competidores	Competidores-Enanos
Enanos	Enanos-Todo
Todo	Todo

En este punto existen dos elementos expuestos: Competidores y Golosos. La forma de elegir a uno de los dos es optar por aquel que sea superclase directa del último elemento de la lista de precedencia, en este caso es Competidores.

Lista de precedencia de clases:

Patriota
Administradores
Competidores

Nodo	Pares de anzuelo
Patriota	~~Patriota-Administradores~~, ~~Administradores-Golosos~~, Golosos-Diaristas
Administradores	~~Administradores-Competidores~~
Golosos	Golosos-Enanos
Diaristas	Diaristas-Enanos
Competidores	~~Competidores-Enanos~~
Enanos	Enanos-Todo
Todo	Todo

Ahora el elemento expuesto es Golosos, agreguémoslo a la lista de precedencia y tachemos sus pares:

Lista de precedencia de clases:

Patriota
Administradores
Competidores
Golosos

Nodo	Pares de anzuelo
Patriota	~~Patriota-Administradores~~, ~~Administradores-Golosos~~, ~~Golosos-Diaristas~~
Administradores	~~Administradores-Competidores~~
Golosos	~~Golosos-Enanos~~
Diaristas	Diaristas-Enanos
Competidores	~~Competidores-Enanos~~
Enanos	Enanos-Todo
Todo	Todo

Ahora el elemento expuesto es Diaristas:

Lista de precedencia de clases:

Patriota
Administradores
Competidores
Golosos
Diaristas

Nodo	Pares de anzuelo
Patriota	~~Patriota-Administradores~~, ~~Administradores-Golosos~~, ~~Golosos-Diaristas~~
Administradores	~~Administradores-Competidores~~
Golosos	~~Golosos-Enanos~~
Diaristas	~~Diaristas-Enanos~~
Competidores	~~Competidores-Enanos~~
Enanos	Enanos-Todo
Todo	Todo

Ahora el elemento expuesto es Enanos y por último Todo.
Lista de precedencia de clases:

Patriota
Administradores
Competidores
Golosos
Diaristas
Enanos
Todo

Nodo	Pares de anzuelo
Patriota	~~Patriota-Administradores~~, ~~Administradores-Golosos~~, ~~Golosos-Diaristas~~
Administradores	~~Administradores-Competidores~~
Golosos	~~Golosos-Enanos~~
Diaristas	~~Diaristas-Enanos~~
Competidores	~~Competidores-Enanos~~
Enanos	~~Enanos-Todo~~
Todo	~~Todo~~

En resumen, cuando un descriptor es afectado por varios procedimientos cuando-se-construye, se utiliza el procedimiento de ordenamiento topológico para encontrar la lista de precedencia de clases, entonces se aplicará el procedimiento más específico de acuerdo a la lista.

Procedimientos demonio

Algunos autores han convenido en llamar *procedimientos demonio* a ciertos procesos que son activados cuando sucede un suceso, se les llama demonio porque mientras tal suceso no ocurra, los procedimientos permanecen acechando. Se señalan los siguientes:

- procedimientos cuando-se-pide
- procedimientos cuando-se-lee
- procedimientos cuando-se-escribe

Los procedimientos cuando-se-pide son utilizados para invalidar datos de una ranura. Por ejemplo:

Cuando se pide un valor para la ranura Apetito de Goloso,

- responda Enorme.

De esta manera, los procedimientos cuando-se-pide manejan una especie de valor de ranura virtual, ya que aunque el dato no esté escrito en la plantilla da la apariencia de que si lo está.

Los procedimientos cuando-se-pide pueden ser tan complicados como se requiera, por ejemplo:

Cuando se pide un valor para el descriptor Peso de un Enano,

- si su descriptor de Apetito es Enorme, responda Grande;
- si no, responda adecuado.

Cuando dos o más procedimientos cuando-se-pide afectan un mismo valor de descriptor, como en el caso de los procedimientos cuando-se-construye, se puede utilizar también el procedimiento de ordenación topológica para determinar la precedencia.

Por otra parte los procedimientos cuando-se-lee y cuando-se-escribe son activados cuando se realizan operaciones de lectura y escritura respectivamente. Un ejemplo de procedimiento cuando-se-escribe es el siguiente:

Cuando se escribe un valor en el descriptor Personalidad de un Enano,

- si el nuevo valor es Tímido, escriba Leer en el descriptor Pasatiempo del Enano.

Como se puede observar, los procedimientos cuando-se-lee y cuando-se-escribe permiten hacer modificaciones a los valores de descriptores de manera automática. A diferencia de los procedimientos vistos anteriormente, todos los procedimientos cuando-se-lee y cuando-se-escribe que afectan a un descriptor se ejecutan, aunque eso si, respetando un orden.

Existen procedimientos llamados *con-respecto-a*, los cuales son procedimientos cuando-se-pide especializados y permiten establecer puntos de vista de acuerdo a un contexto. Por ejemplo:

Cuando se pida el valor del descriptor Humor de Anita en el contexto Estudiando,

- responda Enojada.

Cuando se pida el valor del descriptor Humor de Anita en el contexto Jugando,

- responda Alegre.

El método de descripción y pareamiento

El método de descripción y pareamiento es un método utilizado para la búsqueda y se aplica en diferentes formas de representación del conocimiento entre ellas las plantillas. La idea básica es identificar un objeto describiéndolo primero y luego buscando dentro del conocimiento disponible una descripción que coincida. En español procedimental, el método se describe como sigue:

Para identificar un objeto mediante descripción y pareamiento,

- describa el objeto mediante una representación adecuada;

- compare la descripción del objeto con las descripciones de un acervo hasta que haya un pareamiento satisfactorio o se agoten las descripciones;

- si se encuentra un pareamiento satisfactorio, notifique éxito, de lo contrario notifique fracaso.

Cuando la representación del conocimiento es buena, el desempeño de la búsqueda se optimiza. En ocasiones es posible saber hacia donde dirigir la búsqueda de acuerdo con las diferencias encontradas entre la descripción del objeto y la descripción del acervo.

Conclusiones

Un sistema de plantillas es una forma de representación del conocimiento que utiliza ejemplares, clases, descriptores y valores de descriptores. Sus procedimientos de acceso son los lectores de descriptores y escritores de descriptores, se cuanta además con ciertos procedimientos que permiten manejar la información llamados procedimientos demonio. Los procedimientos cuando-se-construye permiten la aplicación del término herencia en un sistema de plantillas.

Esta forma de representación de conocimiento es considerada como fácil de construir y corregir y los procedimientos descritos anteriormente dan cabida a un proceso de aprendizaje por parte del sistema. Los sistemas de plantillas han tenido auge en los últimos años gracias al advenimiento de los Lenguajes de Programación Orientados a Objetos, como son el C++ o el Lisp y la idea original del concepto se debe a Marvin Minsky en 1975.

I.3

El Proceso de Inferencia

El Motor de Inferencia

El motor de inferencia es la parte del sistema experto que se encarga de realizar conclusiones e inferir nuevos hechos a partir de la base de conocimiento y de la memoria de trabajo; dicho de otra forma, el motor de inferencia es el encargado de generar la solución de acuerdo con la información dada por el usuario y utilizando el conocimiento disponible en el sistema. Para el cumplimiento de su función, el motor de inferencia basa su búsqueda en un método determinado. Cuando la base de conocimiento es tal que el número de posibles soluciones es grande, la necesidad de un método de búsqueda óptimo es patente.

Aunque existe una gran cantidad de estrategias de búsqueda, la mayoría de estas se basa en dos conceptos fundamentales:

a) Encadenamiento hacia adelante.
b) Encadenamiento hacia atrás.

Encadenamiento hacia adelante

También se conoce como *búsqueda guiada por atributos* y es una forma de razonamiento ascendente en la cual se toman las condiciones iniciales y a partir de ellas se trata de generar la solución. Se toman los datos proporcionados por el usuario y se comienza la búsqueda de aquel objetivo que más se acerque a las condiciones dadas. El proceso termina cuando se logra el objetivo o cuando ya se ha analizado toda la información en la base de conocimiento. Por ejemplo, utilizando este método un

médico preguntaría los síntomas de su paciente para después intentar deducir una enfermedad.

Encadenamiento hacia atrás

Es conocido también como *búsqueda guiada por objetivos* y es un proceso de razonamiento descendente en donde a partir de una hipótesis se intenta llegar a las condiciones iniciales del problema. El sistema selecciona un objetivo y dada la información actual verifica que esa sea la solución, si no lo es lo intenta con otro. El proceso termina cuando la hipótesis tomada se cumple o cuando ya no hay información que verificar en la base de conocimiento. Para el ejemplo anterior, utilizando este método, un médico tomaría los síntomas de una enfermedad especifica y comenzaría preguntando si el paciente presenta tales molestias, en caso de no ser así, el médico intentaría con otra enfermedad. Podríamos decir que este método es similar a cuando intentamos solucionar un laberinto partiendo del punto final e intentando llegar al punto de inicio.

Encadenamiento mixto

Es también conocido como búsqueda bidireccional y emplea tanto la búsqueda guiada por atributos como la búsqueda guiada por datos. Aunque el método es muy potente, tiene dos desventajas:

1) Es difícil implementarlo.
2) Es similar a quienes intentando construir un túnel, comienzan a cavar desde ambos extremos sin tener nada que les asegure que se encontrarán en un punto dado.

Además de los anteriores criterios de búsqueda, existen factores a tomar en cuenta para el ahorro de tiempo. Por ejemplo, los sistemas tradicionales realizan búsquedas "a ciegas", es decir, no existe un lineamiento que indique que camino tomar. Los Sistemas Expertos realizan a su vez búsquedas heurísticas. Si bien un heurístico no es un lineamiento infalible, si ofrece en muchas ocasiones una orientación para guiar la búsqueda y ahorrar tiempo. En la siguiente sección se describen algunas técnicas que ayudan a la solución de problemas en los Sistemas Expertos.

METODOS PARA LA SOLUCION DE PROBLEMAS

El Método de Generación y Prueba

Este método consta de dos partes:

a) Generador de soluciones.
b) Probador de soluciones.

La figura 1.8 ilustra este método.

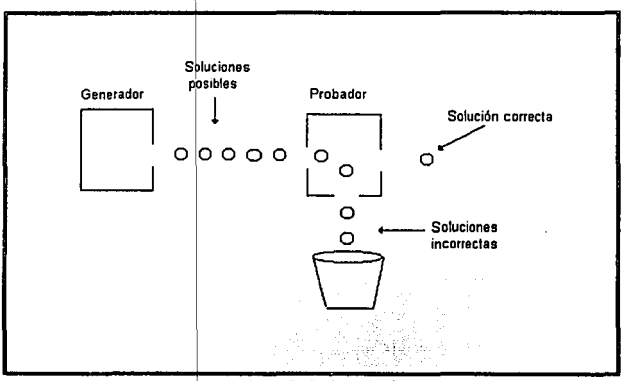

Figura 1.8. *El método de Generación y Prueba.*

El generador proporciona las soluciones posibles al probador y el método termina cuando se encuentra una solución aceptable, cuando se tiene un número satisfactorio de soluciones o bien cuando se hallaron todas las soluciones posibles. En el primer caso, podemos describir el método como sigue:

• Hasta que se encuentre una solución aceptable o no sea posible generar más soluciones:

  • Genere una solución.
  • Pruebe la solución.

• Si se halla una solución satisfactoria menciónela, de lo contrario emita mensaje de fracaso.

Cuando el número de posibles soluciones es grande, para optimizar tiempo se requiere que el generador de soluciones tenga las siguientes características:

a) Ser completo: debe tener la capacidad de generar todas las soluciones que sean posibles.

b) No ser redundante: no debe generar dos o más veces una misma solución.

c) Estar informado: debe contener cierta información que pueda usar para restringir el número de soluciones que pueda generar.

Sobre el último punto podemos decir que de acuerdo a la información que contenga el generador la cantidad de posibles soluciones será mayor o menor y entre más información menos posibles soluciones habrá, por lo que el tiempo del proceso se reducirá también.

En un sistema de identificación de objetos, el proceso de generación consiste en recorrer cada uno de los objetos posibles y el de prueba en la comparación del objeto en turno con el patrón a identificar.

Método de Análisis de Medios y Metas

Este método se basa en el uso de un espacio de estados. El estado de un sistema es una descripción de las condiciones del mismo. Un espacio de estados es un conjunto de estados entrelazados en donde cada enlace representa una posible transición en el paso de un estado a otro.

Cuando hablamos de solución de problemas, identificamos dos estados que son especiales:

- estado actual: es el estado en donde nos encontramos,

- estado meta: es el estado en donde deseamos estar.

La solución del problema se encuentra cuando obtenemos una sucesión de transiciones que nos permita pasar del estado inicial al estado meta.

El propósito del análisis de medios y metas consiste en reducir cada vez más las diferencias entre el estado actual y el estado meta mediante la aplicación de procedimientos que ocasionen transiciones.

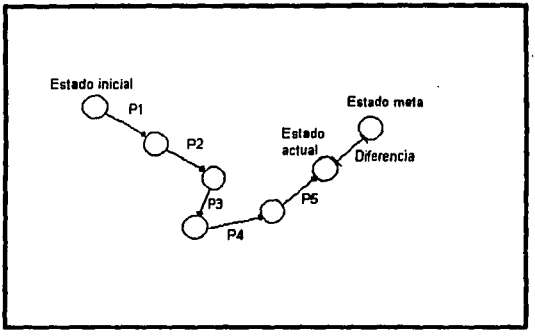

Figura 1.9. Diagrama de estados

En el ejemplo de la figura 1.9 los procedimientos Pi ocasionan transiciones de estado. Aunque la mayoría de las transiciones reducen la diferencia entre el estado inicial y el estado meta, vemos que P3 ocasiona un retroceso. En la forma más básica del análisis de medios y metas, no existen procedimientos integrados que impidan un retroceso en la solución.

Podemos describir este método como sigue:

- Hasta alcanzar la meta o hasta no existir más procedimientos posibles:
  - describa el estado actual, el estado meta y la diferencia entre los dos;
  - en base a la diferencia mencionada seleccione un procedimiento conveniente;
  - utilice el método conveniente y actualice el estado actual.
- Si se alcanza la meta notifíquelo, de lo contrario emita un mensaje de fracaso.

A menudo es conveniente utilizar una *tabla de diferencia-procedimiento* , la cual describe que procedimiento usar de acuerdo a la diferencia entre el estado actual y el estado meta.

METODOS DE BUSQUEDA BASICA

Los métodos descritos en esta sección ayudan al proceso de búsqueda del motor de inferencia. Para realizar una búsqueda óptima tendremos que seleccionar el método más eficiente de acuerdo con la naturaleza de nuestro problema.

Métodos ciegos

Para ejemplificar los métodos de búsqueda, veamos el problema de encontrar una trayectoria de un punto inicial S a un punto final G como se muestra en la figura 1.10.

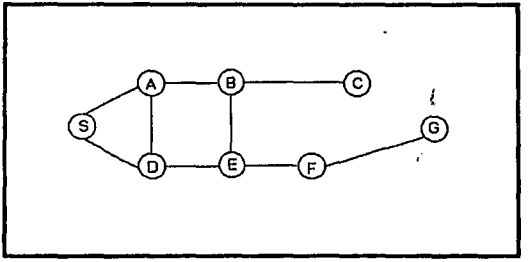

Figura 1.10. *Encuéntrese una trayectoria para llegar del punto S al punto G.*

La forma más fácil de hallar una solución es seleccionar todas las trayectorias eliminando aquellas que son redundantes, por ejemplo S-A-D-S-A-D... Esto se puede hacer por medio de una representación de árbol como la mostrada en la figura 1.11.

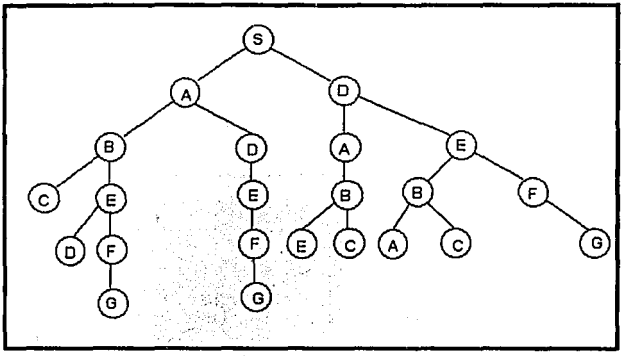

Figura 1.11. *Representación de árbol para el ejemplo de las trayectorias.*

En este árbol se representan las trayectorias. A un nodo que antecede a otro se le llama *padre* y a éste *hijo*. El nodo que no tiene padre se llama *nodo raíz*. Un nodo que no tiene hijos se llama *nodo hoja*.

Cuando un nodo tiene b hijos, se dice que tiene un *factor de ramificación* de b.

Una trayectoria es parcial cuando no alcanza un nodo hoja, de lo contrario es una trayectoria completa.

Se conoce como expansión del nodo al número de hijos que tiene. El número de trayectorias se *expande exponencialmente* a medida que aumenta la profundidad del árbol.

A continuación se describen dos métodos de búsqueda ciegos.

Búsqueda en profundidad

El método de búsqueda en profundidad consiste en realizar la búsqueda recorriendo una trayectoria que parte del nodo raíz hasta llegar a un nodo hoja, de no encontrarse la solución, la búsqueda se continua en el nodo anterior más cercano que tenga una alternativa sin explorar. Se utiliza el criterio de intentar alternativas en el orden de izquierda a derecha.

Para nuestro ejemplo, se muestra el método en la figura 1.12.

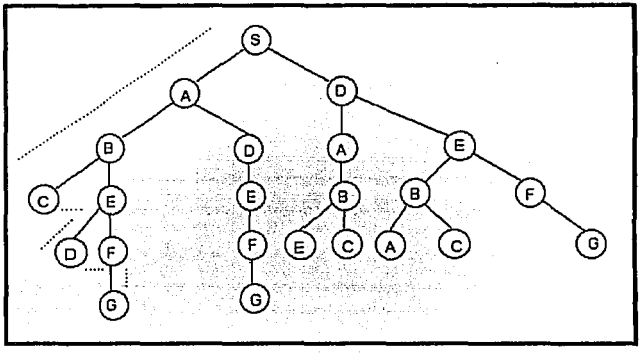

Figura 1.12. La búsqueda en profundidad y con el criterio izquierda-derecha inicia en el nodo raíz y continua hacia abajo hasta llegar a un nodo hoja, enseguida continua desde el nodo anterior más cercano que tenga alternativas sin explorar.

Búsqueda en Amplitud

La búsqueda en amplitud recorre todos los nodos de un mismo nivel antes de avanzar a un nivel más profundo. Para el ejemplo, la búsqueda en amplitud se ilustra en la figura 1.13.

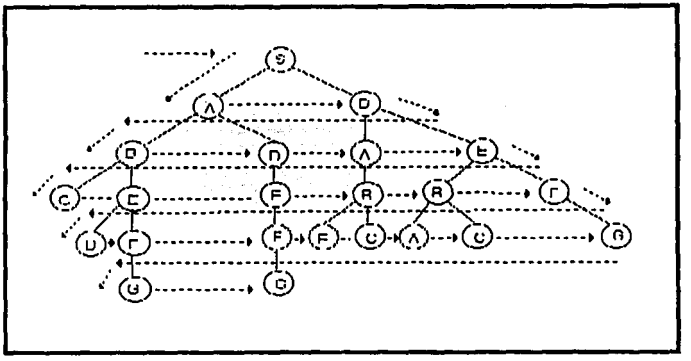

Figura 1.13. *La búsqueda en amplitud recorre todos los nodos de un mismo nivel antes de pasar a un nivel más profundo.*

Conviene usar la búsqueda en profundidad cuando las trayectorias no son muy largas. Así mismo, convendrá utilizar otros métodos de búsqueda cuando las trayectorias sean infinitamente largas. La búsqueda en amplitud funciona bien en árboles cuyas trayectorias son muy profundas y no es recomendable cuando el factor de ramificación es grande.

Existe otro método de búsqueda ciega llamado *Búsqueda Ciega No Determinista,* en dicho método se escoge un nodo al azar y se realiza la búsqueda hasta encontrar la solución o llegar al final de la trayectoria. De esta forma las trayectorias ya analizadas se van desechando y se toman en cuenta únicamente las alternativas que faltan.

Métodos de búsqueda informados heurísticamente

Estos métodos optimizan la búsqueda ya que ésta se hace siguiendo ciertos criterios, de modo que se examinan primero las alternativas más prometedoras. Los

criterios que siguen estos métodos para hacer su búsqueda son llamados heurísticos. A continuación se describen algunos métodos de búsqueda informados heurísticamente.

El Ascenso de Colina

Una forma de desplazarse en un árbol de trayectorias es el ascenso de colina. El ascenso de colina funciona igual que la búsqueda en profundidad, excepto que ordena las alternativas de acuerdo con una medición heurística de la distancia hacia la meta. Mientras mejor sea la medición heurística, mejor será el ascenso en colina. Para nuestro ejemplo, una buena medición heurística sería la distancia en línea recta que hay desde un nodo hasta la meta. Supongamos las distancias mostradas en la figura 1.14:

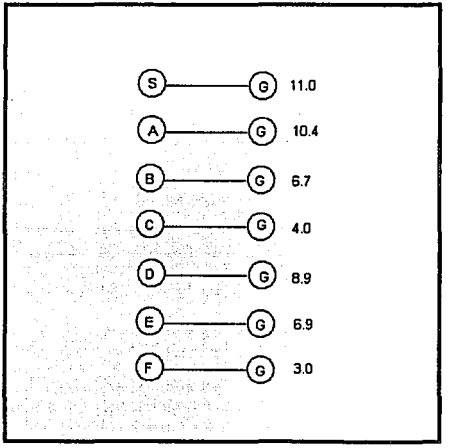

Figura 1.14. Distancias supuestas entre cada punto y la meta.

De acuerdo con lo anterior y haciendo un ascenso de colina, la búsqueda para nuestro ejemplo es la siguiente:

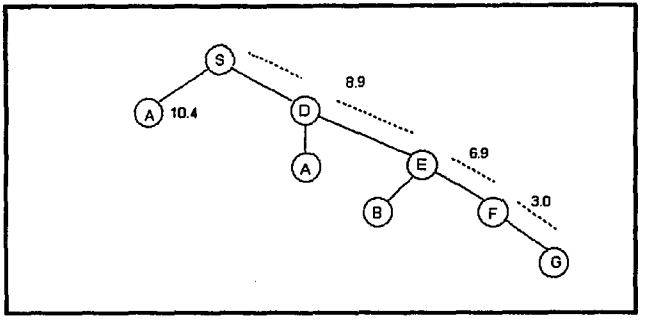

Figura 1.15. *El ascenso de colina es una búsqueda en profundidad con una medición heurística para seleccionar la mejor alternativa. En este ejemplo los números al lado de los nodos indican la distancia en línea recta para llegar de dicho nodo a la meta.*

La búsqueda iniciada en S se dirige hacia D cuya distancia es menor que A, enseguida va hacia E en preferencia al nodo A cuya distancia es mayor; estando en E se elige F y por último se llega a G.

Como podemos observar, entre más conocimiento se tenga para la búsqueda, más tiempo se ahorra.

La Búsqueda en Haz

Es similar a la búsqueda en amplitud en cuanto a que realiza la búsqueda en forma transversal en nodos del mismo nivel, sin embargo selecciona únicamente w nodos para recorrerlos. De esta manera se mantiene un control de los nodos aún cuando la ramificación sea grande y las ramas sean profundas. La búsqueda en haz selecciona los w mejores nodos o los más prometedores para continuar la búsqueda y no toma en cuenta los demás.

La Búsqueda Primero el Mejor

En este método se selecciona el nodo que más cercano esté a la meta sin importar la posición que tenga en el árbol parcialmente desarrollado. En muchas ocasiones este método llega a la solución empleando trayectorias más cortas en relación a los demás métodos de búsqueda.

En ocasiones no es posible tener una medida natural sin antes sensar cada una de las alternativas para seleccionar la mejor; podría ser necesario caminar a cada una de las alternativas y continuar la búsqueda en aquella que ofrezca un mayor acercamiento a la solución y un menor esfuerzo.

Conclusiones

Podemos resumir como sigue los métodos de búsqueda básica:

Procedimientos ciegos:
- Búsqueda en profundidad
- Búsqueda en amplitud
- Búsqueda no determinista

Procedimientos informados heurísticamente:
- Ascenso de colina
- Búsqueda en haz
- Búsqueda primero el mejor

Cada uno de los métodos anteriores es bueno bajo ciertas condiciones del árbol de búsqueda. Se recomienda usar la búsqueda primero en profundidad cuando las trayectorias no sean muy largas; la búsqueda en amplitud es buena cuando el factor de ramificación no es muy grande, y si no se tiene la certeza de cuál de las dos búsquedas, en profundidad o en amplitud, es mejor se recomienda usar la búsqueda no determinista.

Recordemos que mientras mayor conocimiento se tenga para hacer la búsqueda, podemos ahorrar más tiempo, por lo que si se tiene una medida natural de la distancia de cada nodo a la meta, úsese alguno de los procedimientos informados heurísticamente.

I.4

Los Módulos de Comunicación

MODULOS DE COMUNICACION

Como se mencionó en un principio, existen dos módulos de comunicación (o interfaces) en un sistema experto: el módulo de comunicación del experto y el módulo de comunicación del usuario. Básicamente estos dos módulos permiten la comunicación del sistema con el hombre en dos niveles, en el nivel de usuario y en el nivel de experto.

Se dice que un módulo de comunicación hombre-máquina debe ser:

- Rápido, de manera que la comunicación no sea tediosa para el usuario.
- Potente, es decir, que maneje estructuras lo más cercano a un lenguaje natural.
- Sencillo, de manera que sea entendible para el usuario.

Los módulos de comunicación se valen de texto, gráficos, animación, iconos y color, sistemas multiventanas y ayuda en línea para la realización de sus funciones.

Módulo del usuario

Se conoce también como interface usuario-sistema experto y es el que hace posible la comunicación entre ambos de la manera más natural posible.

La comunicación entre el usuario y el sistema es necesaria para las siguientes tareas:

- Entrada de datos.
- Elección de opciones del sistema.
- Salida de explicaciones y justificaciones.
- Salida de soluciones.

En muchos sistemas expertos el usuario contesta preguntas mediante menús que presentan opciones múltiples lo cual hace que la entrada de datos sea rápida y sencilla. El módulo del usuario interviene también en el proceso de inferencia en el caso en el que el motor de inferencia solicite algún dato para la obtención de una solución. Y por último, cuando se completa el proceso de razonamiento, el módulo del usuario presenta el correspondiente resultado en pantalla.

En el caso en que no se halle solución, el módulo del usuario podría no presentar conclusión alguna en la pantalla o bien podría presentar una aproximación a la solución. En otras ocasiones se presenta una solución acompañada de un factor de certidumbre que la califique.

Existen sistemas expertos cuyos módulos del usuario utilizan el lenguaje natural. Dichos módulos emplean las técnicas de la Inteligencia Artificial para recibir información del usuario en lenguaje natural y transportarla a formatos compactos utilizables por la computadora, así mismo dan sus respuestas en lenguaje natural. Sin embargo, son pocos los sistemas expertos que usan este tipo de módulos ya que el construirlos tiene el mismo grado de dificultad que construir el sistema experto. Se justifica el uso de lenguaje natural en un sistema experto cuando éste es de gran tamaño.

Componente explicativo

Una de las tareas más laboriosas del módulo del usuario es la explicación y justificación de soluciones, la cual es realizada por un elemento llamado componente explicativo.

El componente explicativo presenta el lineamiento seguido para llegar a una conclusión. La razón de la explicación de una solución es que al presentar al usuario los motivos del razonamiento éste pueda confiar en ella. La explicación puede ser un método de depuración para el humano al analizar el camino que se siguió. La explicación es importante sobre todo cuando se trata de tomar decisiones críticas.

Lo que hace el componente explicativo es mostrar cada una de las partes del conocimiento que aplicó y la forma en que fue construyendo la solución. Existen aplicaciones en las que el componente explicativo no es necesario puesto que las soluciones deben aplicarse al instante y no hay tiempo de verificar dicha solución.

Módulo del experto

Las funciones que realiza el módulo de comunicación con el experto son las siguientes:

a) Adquisición de conocimiento: permite la inclusión de conocimiento del sistema experto en la base de conocimiento. En primera instancia, este submódulo se utiliza en la construcción del sistema. Posteriormente es utilizado para mantener actualizado el conocimiento, permitiendo la edición del conocimiento ya existente y la adición de nuevo conocimiento. Generalmente este subsistema es un editor de textos especializado que realiza la actualización de la base de conocimiento de la misma forma que un compilador da a un texto el formato apropiado. La actualización del conocimiento es muy importante en áreas donde éste cambia constantemente, ya que de no hacerlo las soluciones del sistema experto pierden vigencia y llegan a ser equivocadas.

b) Validación y depuración del conocimiento: permite detectar repeticiones, inconsistencias y errores.

c) Configuración del sistema: permite configurar el motor de inferencia y los módulos de comunicación a los requerimientos del usuario. Cuando el sistema experto se desarrolla en un lenguaje de alto nivel, este submódulo es el editor del lenguaje

Conclusiones

Los módulos de comunicación son aquellos que permiten el intercambio de información entre el ingeniero de conocimiento y el sistema experto o bien entre el usuario y el sistema experto. Se auxilian de ventanas, menus, colores y en ocasiones emplean lenguaje natural. Los módulos de comunicación deben ser rápidos, sencillos y entendibles.

I.5

El Proceso de Construcción de un Sistema Experto

EL PROCESO DE CONSTRUCCION DE UN SISTEMA EXPERTO

Partiendo del punto de que un sistema experto es un sistema de software, las metodologías para el desarrollo de software tradicional son muy parecidas a las de desarrollo de sistemas expertos. Los métodos de desarrollo de sistemas expertos han aprovechado las bases de la Ingeniería de Software para el desarrollo de sistemas tradicionales.

Existen diez pasos básicos en la construcción de un sistema experto:

1. Definir el problema.
2. Evaluar soluciones alternativas.
3. Comprobar la viabilidad de una solución basada en el uso de sistemas expertos.
4. Estimación de las inversiones y beneficios.
5. Elección de una herramienta de desarrollo.
6. Aplicación de la Ingeniería del Conocimiento.
7. Desarrollo de la base de conocimiento.
8. Desarrollo del software.
9. Comprobación y validación del sistema.
10. Mantenimiento del sistema.

I . La Teoría de los Sistemas Expertos

1. Definición del problema

En el desarrollo de cualquier software este es el primer paso: definir el problema. Antes de comenzar a invertir en dinero y esfuerzo, es importante tener una visión clara del problema. Se deben plantear las preguntas ¿cuál es el problema? y ¿cuál es la necesidad real? Pueden existir problemas de baja productividad, falta de expertos, mal acceso a la información, etc.

Para definir el problema es recomendable escribir una relación clara de las necesidades existentes; dicha observación podría incluir observaciones de la gente que realiza el trabajo, entrevistas con los usuarios, etc.

De acuerdo con la definición del problema se podrá decidir si se requiere un sistema experto o si la mejor solución es un sistema tradicional.

2. Evaluación de soluciones alternativas

Los problemas que pueden solucionar los sistemas expertos están relacionados con la falta de conocimiento. Sin embargo, existen otras alternativas que pueden solucionar un problema de falta de conocimiento. Antes de iniciar la construcción de un sistema experto, se deben considerar soluciones alternativas como las siguientes:

a) *Facilitar la consulta a expertos humanos:* si existe una falta de acceso al conocimiento, una solución simple sería contactar a un experto humano. Posibles soluciones en este sentido serían establecer comunicación con un experto humano, contratarlo o formar nuevos expertos.

b) *Capacitación y formación:* otra forma de solucionar el problema es proporcionar a una persona la instrucción y educación que requiere para desempeñar su trabajo. A largo plazo esta es una buena solución, sin embargo, hay dos inconvenientes: el tiempo que se tarda toda la capacitación y el hecho de que cuando una persona se va de la empresa, se lleva su conocimiento. Un sistema experto permite conservar los conocimientos e incluso servir como medio de capacitación para nueva gente.

c) *Impresión del conocimiento:* se refiere a la elaboración de manuales y textos de consulta del conocimiento. De esta forma la información puede ser accesada fácil y rápidamente. Aunque el proceso de elaboración de manuales y textos es tardado, el costo es menor al de desarrollar un sistema experto.

d) *Software convencional:* cuando ya se determinó que la solución por computadora es la mejor para el problema, hay que decidir entre un sistema convencional y un sistema experto. Pudiera ser que el problema se resolviera con alguno de los

paquetes comerciales existentes o bien con el desarrollo de un sistema convencional. Recuerde que es importante elegir un medio que nos permita ahorrar esfuerzos y obtener una solución aceptable.

3. Comprobación de la viabilidad de una solución basada en el uso de un sistema experto

Para estar seguros de que un sistema experto es la mejor solución a nuestro problema, nuestra aplicación debe caer en alguna de las áreas de aplicación de los sistemas expertos, las cuales son:

1. Análisis e interpretación
2. Predicción
3. Diagnóstico y depuración
4. Control
5. Diseño
6. Planificación
7. Enseñanza

Si se cae en alguna de las categorías anteriores, ahora el problema debe cumplir también con las siguientes condiciones:

a) *Necesidad de expertos con los conocimientos adecuados:* un problema que no requiera de un experto humano para ser solucionado, no es una buena aplicación para un sistema experto. Cuando un problema se soluciona con la ayuda de un experto, un sistema experto es una buena alternativa cuando se carezca de expertos o simplemente se desea conservar el conocimiento de un experto que se jubila.

b) *Disponibilidad de expertos:* para el desarrollo de un sistema experto es indispensable contar con uno o más expertos humanos dispuestos a aportar su conocimiento.

c) *Un campo limitado:* el campo en el que se va a aplicar un sistema experto debe ser perfectamente limitado y reducido. Un sistema experto no puede actuar sobre generalidades ya que sería muy extenso. Por el contrario, un problema muy específico si puede resolverse con un sistema experto.

d) *La naturaleza del conocimiento:* el conocimiento que se maneje debe ser adaptable a alguna de las formas de representación del conocimiento: reglas de producción, cálculo de predicados, plantillas, etc.

e) *Alta probabilidad de recuperar las inversiones:* el desarrollo de un sistema experto requiere de una buena cantidad de dinero y recursos, por lo que antes de iniciar la construcción de uno se debe estar seguro que los beneficios serán mayores que el costo.

f) *No necesita aplicar el sentido común:* existe una gran cantidad de problemas que se resuelven con sentido común. Por ejemplo, se que si toco una plancha caliente me quemo. Dado que la aplicación de sentido común implica el conocimiento de muchas áreas no entrelazadas, un sistema experto no puede resolver un problema por sentido común. El problema a resolver debe requerir conocimiento, juicio y experiencia.

g) *Dificultad de la solución:* el problema debe tener solución. No debe resolverse mediante una simple consulta a manuales o libros. Las soluciones del problema no deben requerir una cantidad muy limitada de conocimiento y no deben ser muy sencillas. Tampoco deben ser excesivamente complejas ya que el sistema experto podría no estar a la altura del problema. Un sistema experto es capaz de resolver problemas de dificultad moderada que se encuentran entre ambos extremos.

h) *Un número razonable de respuestas posibles:* en el desarrollo de un sistema experto se deben generar todas las posibles soluciones al problema, de manera que el sistema encuentre el camino hacia alguna de ellas. Cuando el número de soluciones es muy reducido (2 o 3) no tiene sentido construir un sistema experto; de la misma forma un problema con cientos o miles de posibles soluciones es muy complejo para un sistema experto. Algunos autores señalan que para que un sistema experto sea funcional debe tener como límite máximo 100 soluciones.

i) *No existen algoritmos definidos:* cuando el problema se soluciona mediante la aplicación de un algoritmo, la solución de ecuaciones o la aplicación de un modelo estadístico, la construcción de un sistema experto está de sobra.

4. Estimación de las inversiones y beneficios

Como todo el desarrollo de software, la construcción de un sistema experto requiere ciertos recursos. Para que el empleo de un sistema experto sea un éxito, los beneficios deberán se mayores a los costos. Para determinar lo anterior es necesario hacer un análisis costo-beneficio; a continuación se muestran algunos puntos para ello.

Estimación de los costes de desarrollo

En los gastos que se tendrán al implementar un sistema experto debemos considerar primero qué herramientas de desarrollo de software se necesitarán y en su caso si se requerirá nuevo equipo de hardware, monitores o algún periférico.

Enseguida habrá que considerar los honorarios de la gente que trabajará en el proyecto. En general, un proyecto de desarrollo de un sistema experto requiere de las siguientes personas:

- *El experto:* es la persona o grupo de personas que aportará el conocimiento para la solución del problema.

- *El ingeniero de conocimiento:* es el responsable de transportar el conocimiento a un formato entendible por la computadora, en otras palabras es quien diseña la estructura para la representación del conocimiento. Debe reunir las siguientes características:

  - Espíritu abierto, debe tener la disposición para considerarse ignorante en muchas áreas del conocimiento y poder asimilar las exposiciones de los expertos.

  - Conocimientos multidisciplinarios que le permitan abordar problemas de dominios diversos.

  - Capacidad de organización, para poder organizar grandes cantidades de información.

  - Curiosidad intelectual, para poder profundizar en los problemas y sus soluciones.

- *El programador:* debe ser una persona con conocimiento en el desarrollo de software y es el encargado de instrumentar a nivel programación el diseño logrado por el ingeniero de conocimiento y el ingeniero en computación.

- *El ingeniero en computación:* es el encargado de elegir o diseñar el software y hardware requerido para la construcción del sistema experto.

Es claro que en el desarrollo de un sistema experto pequeño alguna de las personas antes mencionadas podría cubrir varias funciones e incluso un experto con conocimientos de computación podría hacer todo el trabajo. Sin embargo, en el desarrollo de un sistema grande se requerirán al menos dos personas: el experto y el ingeniero de conocimientos.

Para la estimación de los costes es necesario considerar los honorarios del equipo de trabajo y tomar en cuenta el tiempo de desarrollo, siempre es bueno añadir algunos días a nuestra estimación recordando que la mayoría de las veces se presentan improvistos.

Estimación de los beneficios

Estimar los beneficios que trae un sistema experto es un proceso difícil. Cuando el sistema experto resuelve un problema que genera pérdidas cuyo monto se conoce, se tiene ya un punto de partida. De otra forma, habrá que comenzar el análisis desde cero. Los beneficios de un sistema experto podrían ser algunos de los siguientes:

- Realizar más trabajo en menos tiempo, obteniendo mayores rentas y mejorando la productividad.
- Reducir pérdidas de tiempo o material.
- Mayor exactitud en soluciones aumentando la confianza de los clientes.

Solo cuando los beneficios superan a los costes, el sistema experto debe ser implementado. También se debe considerar que el sistema experto es desarrollado una sola vez y que sus beneficios pueden durar varios años, por lo que se puede amortizar los costes de desarrollo a lo largo de varios años para mejorar los beneficios de la inversión.

5. Elección de una herramienta de desarrollo

Se conoce como herramienta de desarrollo a un conjunto de programas que permiten la creación de otros programas. La elección de una herramienta depende de varios factores:

- recursos de programación disponibles,
- lenguajes disponibles,
- hardware disponible,
- la naturaleza del problema.

Las diferentes herramientas para el desarrollo de un sistema experto pueden clasificarse en tres grupos:

a) Lenguajes de programación convencionales.
b) Lenguajes de Inteligencia Artificial.
c) Shells.

FACILIDAD DE USO
⟶

LENGUAJES DE PROGRAMACION CONVENCIONALES			LENGUAJES DE PROGRAMACION DE INTELIGENCIA ARTIFICIAL		
ENSAMBLA-DORES	LENGUAJES ORIENTADOS AL PROCEDI-MIENTO (FORTRAN, BASIC, C, PASCAL...)	LENGUAJES ORIENTADOS AL OBJETO (C++, SMALLTALK)	LISP	PROLOG	SHELLS

⟵
TIEMPO DE DESARROLLO

Tabla 1.16. *Herramientas para el desarrollo de un sistema experto*

La tabla 1.16 esquematiza las diferentes herramientas para el desarrollo de un sistema experto. Se observa de acuerdo a las flechas que los ensambladores son los más difíciles de usar y los shells los más fáciles, asimismo se tarda más tiempo desarrollar un sistema experto en ensamblador y menos tiempo en un shell.

A continuación se muestran algunas características de dichas herramientas.

Lenguajes convencionales

Se han creado sistemas expertos en los lenguajes de programación más comunes: Basic, Pascal, Fortran, C. Se recomienda utilizar alguno de estos lenguajes cuando se tiene conocimiento y experiencia en él ya que es más tardado aprender otro lenguaje.

Después de Prolog y Lisp, los lenguajes más utilizados en Inteligencia Artificial son los estructurados Pascal y C. Actualmente se desarrollan muchos trabajos en C por su flexibilidad y porque produce un código rápido.

Uno de los factores importantes en un sistema experto es su velocidad de ejecución. Un intérprete de Basic es muy lento por lo que no debe usarse para sistemas grandes. En el otro extremo, el lenguaje ensamblador produce un código muy compacto y rápido. Los lenguajes Pascal, Fortran y C generan un código rápido.

Desarrollar un sistema experto en un lenguaje de programación convencional es tardado porque hay que construir cada una de las partes del mismo, a diferencia de otras herramientas como el Prolog.

Otros de los lenguajes que están utilizándose mucho en Inteligencia Artificial son los que están orientados al objeto, tal como el C++ o Smalltalk, los cuales se adaptan a representaciones del conocimiento como las redes semánticas o las plantillas.

Lenguajes de programación para la Inteligencia Artificial

Existen lenguajes que fueron creados específicamente para el desarrollo de aplicaciones de Inteligencia Artificial, los más conocidos son el Lisp y el Prolog. Con este tipo de lenguajes el desarrollo de un sistema experto es más rápido que con los lenguajes convencionales. A continuación se describe un poco más el Lisp y el Prolog.

a) Lisp:

Es el lenguaje más utilizado en aplicaciones de Inteligencia Artificial. Existen versiones de Lisp para casi cualquier tipo de computadora. Lisp es un acrónimo de List Processing (tratamiento de listas) y fue desarrollado en los años 50 por John McCarthy. Actualmente el uso de Lisp es tan importante que se han creado computadoras especiales llamadas "máquinas Lisp" para implementar el lenguaje.

Lisp es un lenguaje de programación simbólico que representa la información en forma de listas, también denominadas expresiones. Lisp proporciona diversos métodos para manipular las listas y aunque la mayoría de las operaciones son simbólicas, también puede realizar operaciones aritméticas básicas. Lisp permite manejar prácticamente todos los esquemas de representación del conocimiento. Los autores

califican a Lisp como un lenguaje potente, flexible y muy utilizado, es considerado como el Lenguaje de la Inteligencia Artificial.

b) Prolog:

Es un lenguaje de Inteligencia Artificial creado en Francia en los años 70 y usado ampliamente en Europa y Japón. A diferencia de Lisp, Prolog no es un lenguaje de tipo procedimental sino declarativo. Un lenguaje es procedimental cuando en las sentencias del programa se indican los pasos a seguir para resolver el problema siguiendo un algoritmo dado. En un lenguaje declarativo como Prolog, se describen solo un conjunto de hechos y reglas que describen sus relaciones. Prolog es una implementación de procedimientos para la manipulación de información. Prolog toma su nombre de la expresión PROgramming in LOGic (programación en lógica). Prolog es la herramienta ideal para el desarrollo de un sistema experto. Provee un motor de inferencia, que en la mayoría de los casos emplea el encadenamiento hacia atrás, por lo que sólo es necesario implementar la base de conocimiento.

Shells

Un shell de un sistema experto es un conjunto de programas que permiten crear un sistema experto sin la necesidad de programar. El shell contiene el motor de inferencia, la interfaz de usuario con su componente explicativo y generalmente dispone de un medio adecuado para accesar la base de conocimiento. De esta forma, para construir un sistema experto con un shell solo hace falta desarrollar la base de conocimiento. Los shells simplifican y aceleran el desarrollo de un sistema experto y puesto que no requieren trabajos complicados de programación, los costos también disminuyen. De acuerdo con la forma de implementar la base de conocimiento, existen dos tipos de shells:

a) Shells basados en el uso de reglas: basan su representación del conocimiento en el uso de reglas de producción. Cada shell tiene su propio formato para incorporar una regla del tipo SI-ENTONCES a la base de conocimiento. Las reglas son introducidas utilizando un editor de textos. Algunos shells contienen un editor de reglas, que no es más que un programa de tratamiento de textos sencillo que permite capturar las reglas en un archivo ASCII con un formato determinado, el cual después es compilado para el proceso de inferencia. Algunos shells basados en el uso de reglas permiten la creación de plantillas y redes semánticas. Ejemplos de este tipo de shells son: VP-Expert, XSYS, Turbo Expert, etc.

b) Shells basados en sistemas de inducción: en este tipo de shells se construye la base de conocimiento capturando la información en forma de hechos en una plantilla similar a la de una hoja de cálculo. Se forma una matriz en donde se definen los valores de las variables y el resultado de dichos valores. El shell forma entonces reglas IF-THEN de esa tabla. Por ejemplo, la siguiente regla para el diagnóstico de una regadera de agua se ilustra en la fila 2 de la tabla 1.17:

SI	la regadera está averiada
y	gotea la llave
y	la tuerca de empaque está floja
ENTONCES	aprieta la tuerca de empaque

	REGADERA	LLAVE	TUERCA	RESULTADO
1:	averiada	gotea	ok	cambia_empaque
2:	averiada	gotea	floja	aprieta_tuerca

***Tabla 1.17.** Matriz de ejemplos en un shell de inducción.*

Este tipo de shells facilita de esta forma aun más la construcción de la base de conocimiento. Ejemplos de shells de inducción son: TIMM, Expert Ease, KDS y Rule Master.

Antes de adquirir un shell es necesario considerar las siguientes características:

a) *Entorno operativo:* es la especificación del ambiente en el cual trabaja el shell, incluye el modelo de computadora, el sistema operativo y cualquier otro aspecto importante sobre software o hardware, por ejemplo: memoria RAM necesaria, desde que unidades de disco se accesa, etc.

b) *Tipo de shell:* indica si es de inducción o basado en reglas. Recuerde que los sistemas basados en reglas también permiten otro tipo de representación del conocimiento como las redes semánticas y las plantillas. Algunos sistemas de inducción permiten escribir reglas directamente.

c) *Lenguaje del shell:* es el lenguaje en el que fue desarrollado el shell. La mayoría de los shells están desarrollados en Pascal y C. Otros muchos fueron hechos en Prolog o Lisp y algunos en Basic o Fortran. Los shells hechos con un intérprete de Prolog o Lisp requieren mayor cantidad de memoria para tener residente su intérprete a diferencia de los programas compilados que ocupan menos memoria.

d) *Sistema de control:* se refiere al método empleado por el motor de inferencia. La mayoría de los shells usan uno de los dos métodos: encadenamiento hacia atrás o encadenamiento hacia adelante; pero existen otros que permiten al usuario elegir uno de los dos. De acuerdo a la naturaleza del problema se debe determinar qué método es el más adecuado y adaptar el shell. También se debe considerar si el shell hace un buen manejo de incertidumbre.

e) *Capacidades matemáticas:* es importante verificar qué operaciones matemáticas puede hacer el shell cuando nuestra aplicación requiere que se haga cálculos con los datos de entrada.

f) *Conexiones con otros productos de software:* en ocasiones es deseable que el shell pueda intercambiar datos con algunos paquetes como hojas de cálculo, bases de datos o importar y exportar gráficas.

g) *Lenguajes de programación:* algunos shells vienen acompañados de algún lenguaje de programación de manera que se puede complementar con rutinas desarrolladas en dicho lenguaje.

h) *Interfaz con el usuario:* aquí hay que ver la forma en la que el sistema se comunica con el usuario: si lo hace a través de menús, ventanas, usa color, etc.

i) Otras consideraciones son la versión del shell, el precio, el soporte técnico, la documentación y las regalías, las cuales son cuotas que exige el fabricante del shell para la implementación del sistema experto.

Al elegir un shell cuando existen tanto uno basado en reglas como uno de inducción, elija el de inducción ya que el desarrollo es más fácil y rápido. Considere también el límite que maneja como máximo el shell.

6. Aplicación de la Ingeniería del Conocimiento

Los pasos anteriores en el desarrollo de un sistema experto componen el proceso de planificación y preparación. En esta etapa comienza el trabajo real de desarrollo. *Ingeniería del conocimiento* es el proceso mediante el cual se extrae la información de los expertos humanos y se conforma la base de conocimiento. La persona encargada de este proceso es conocida como *ingeniero se conocimiento* y es alguien con conocimientos de Inteligencia Artificial y de Informática. Su principal habilidad es extraer el conocimiento de los expertos humanos y algunas veces tiene capacidad para programar.

Las fuentes de conocimiento son principalmente dos: la literatura y los expertos.

I . La Teoría de los Sistemas Expertos

a) La literatura.

Entendamos por literatura toda la documentación existente sobre el problema, aquí hablamos de libros, revistas, manuales, informes, etc. Se recomienda preguntar a los expertos qué literatura técnica emplean ellos. Una vez adquirida dicha literatura, hay que revisarla y conocer el contenido para utilizarlo una vez que se requiera. Es deseable que el ingeniero de conocimiento se familiarice con el problema identificando términos empleados y conociendo el entorno mediante la literatura disponible.

b) Entrevistas con los expertos.

Si bien es importante contar con literatura como fuente de conocimiento, el contar con un experto para desarrollar el sistema es indispensable, recordemos que son sus conocimientos los que serán empleados por el sistema. No se puede utilizar únicamente el conocimiento impreso, hace falta un factor: la experiencia personal.

El disponer de uno o varios expertos para el desarrollo de software puede ser algo complicado. Generalmente ellos no disponen de tiempo por lo que será un verdadero privilegio si se cuenta con un experto asignado exclusivamente al desarrollo del sistema experto. Además el o los expertos que colaboren deben estar interesado en hacerlo, de manera que no se muestren hostiles al cooperar. Algunos problemas que se presentan en la comunicación del ingeniero de conocimiento con el experto son los siguientes:

- Cuanto más experta es una persona, más trabajo le cuesta expresar su conocimiento.
- Por lo general el experto no sabe de sistemas expertos ni el ingeniero de conocimiento sabe del área del problema, lo cual dificulta las relaciones entre ambos.
- Aun de forma inconsciente, el experto puede creer que el sistema experto le sustituya o le relegue a tareas menos creativas por lo que puede mostrar una tendencia a no colaborar plenamente con el proyecto.

Se dice que es mejor trabajar con un solo experto que con varios. Esto es debido a que cada persona tiene su forma particular de resolver un problema y al presentar lineamientos de resolución diferentes, el ingeniero de conocimiento tiene problemas al tratar de integrarlos. Cuando es necesario trabajar con varios expertos, es bueno hacerlo por separado y otorgarle a uno de ellos cierta responsabilidad para adaptar todas las corrientes surgidas en un solo esquema del conocimiento.

Generalmente el ingeniero de conocimiento consulta al experto durante cierto tiempo, recogiendo poco a poco la información. El propósito de las entrevistas es recolectar el conocimiento necesario e identificar el proceso de razonamiento para la solución del problema.

En las entrevistas se toman notas sobre las cuestiones planteadas y en ocasiones se usan grabadoras para no tener que repetir las preguntas.

Algunas formas de entrevistas empleadas son las siguientes:

1) Observación, el ingeniero de conocimiento observa como el experto soluciona un problema y si es posible también él participa. El ingeniero de conocimiento anota el proceso seguido para su posterior análisis.

2) Planteamiento de preguntas y problemas a los que ha de responder el experto.

3) Cuando el ingeniero de conocimiento ya se ha adentrado en el área, pregunta por conocimientos que sabe de antemano que necesita.

4) En algún momento durante el proceso de entrevistas, el ingeniero de conocimiento podrá generar algunas reglas y llevarlas al experto para que diga si son válidas o no y por qué.

7. Desarrollo de la base de conocimiento

Aunque esta etapa es parte de la aplicación de la ingeniería del conocimiento, aquí se trata aparte. Una vez obtenido el conocimiento, habrá que organizarlo de manera que se pueda traducir a alguna de las formas de representación del conocimiento. Para ello es necesario pasar por las siguientes etapas:

a) Definición de las soluciones: un sistema experto no genera soluciones por espontaneidad, por lo cual el ingeniero de conocimiento debe generar todas las posibles soluciones, respuestas, preguntas y recomendaciones que el sistema experto dará. Para esto, en las sesiones con los expertos se debe haber conocido una variedad de casos de manera que se hayan contemplado todas las soluciones existentes.

En un sistema de reglas de producción, esta etapa del proyecto nos lleva a construir la parte THEN de las reglas y en sistemas pequeños no se requiere de un árbol de decisión por lo que el sistema solo va pidiendo datos que satisfagan la parte IF de las reglas para encontrar una respuesta.

b) Definición de los datos que hay que suministar al sistema: estos son todos los datos que necesita el sistema experto para realizar la inferencia. Son todas las preguntas que el sistema experto hará al usuario. En un sistema de producción, estos datos son la parte IF de las reglas. El contar con los mismos, hará posible el llegar a una conclusión.

c) Desarrollo de una jerarquía: una de las formas de organizar mejor la información es el desarrollar una jerarquía. Todo el conocimiento puede ser clasificado de alguna forma y generalmente se puede organizar la información en un diagrama de bloques similar a un organigrama. Esto facilita el manejo de información una vez que se tienen los posibles resultados y los datos de entrada necesarios.

d) Construcción de un árbol de decisión: en ocasiones el conocimiento es de tal naturaleza que se puede organizar directamente un árbol de decisión en vez de una jerarquía. La idea de tener un árbol de decisión es el contar con una guía en la búsqueda. Cuando la base de conocimiento es muy grande es complicado hacer un árbol de decisión para toda la base, por lo que se recomienda hacer árboles para distintos subconjuntos de la base de conocimiento.

e) Preparación de una matriz: como vimos en los shells, es bueno representar el conocimiento como una matriz de ejemplos. Cuando utilice un shell de inducción, este paso es indispensable, cuando no, esta es una buena forma de organizar el conocimiento en papel antes de comenzar con el desarrollo de software.

8. Desarrollo del software

Una vez desarrollada la base de conocimiento se puede comenzar a implementar el sistema experto. Se recomienda la construcción de un prototipo que contenga un subconjunto del conocimiento y al cual se le puedan hacer pruebas con el propósito de corregir cualquier deficiencia. En sistemas de representación del conocimiento en donde es posible la modularidad, se puede ir desarrollando el sistema por partes para integrarlo al final; en caso de errores en alguno de los módulos la detección y la corrección son fáciles de hacer.

9. Comprobación y validación del sistema

Una vez obtenido el prototipo, debe someterse a ciertas pruebas. Se recomienda que lo utilicen los expertos y comprueben la validez de las soluciones, así mismo algunas de las personas que van a ser usuarios del sistema. El objetivo de una etapa de pruebas es encontrar deficiencias en el sistema, si hace lo que debe de hacer y si resuelve los problemas para los cuales fue creado. El proceso de pruebas puede repetirse varias veces hasta que no se encuentre ninguna falla. Esta forma de

retroalimentación es importante para lograr un sistema experto con soluciones aceptables. Los grandes sistemas expertos pasan por varias etapas de prueba por lo que se tienen las siguientes versiones:

Etapa	Función
Prototipo de demostración	Demuestra la viabilidad utilizando un subconjunto.
Prototipo de investigación	Una primera versión operacional del sistema completo pero sin probar.
Prototipo de trabajo	Una versión comprobada y fiable
Modelo para producción	Demuestra buenas presentaciones en el entorno del usuario
Sistema comercial	Versión final para uso o venta

Tabla 1.18. *Etapas en el desarrollo de un sistema experto.*

Las principales fallas que se pueden encontrar en un sistema experto son las siguientes:

a) Fallas en la base de conocimiento: esto es, la información contenida en la base de conocimiento es falsa o incorrecta. Este tipo de falla es fácilmente corregible cuando la representación del conocimiento es entendible.

b) Fallas en el motor de inferencia: estas se presentan cuando el conocimiento es correcto pero su aplicación es incorrecta. Se detecta mediante pruebas hechas al sistema experto, sin embargo, en sistemas grandes es difícil agotar todas las posibles combinaciones por lo que el proceso de validación es tardado.

10. Mantenimiento del sistema

Existe una frase que dice: "un sistema experto es un sistema vivo". Esto quiere indicar la necesidad de actualización que tienen los sistemas expertos. Un sistema

experto puede ser útil por un cierto periodo de tiempo, sin embargo, si no se actualiza puede convertirse en obsoleto e incluso inútil. Esto sucede con casi cualquier sistema de software, se van modificando de acuerdo al cambio de necesidades. Al proceso de actualización de los sistemas expertos se le llama *mantenimiento.* Cuando el sistema está hecho en algún shell, la actualización es sencilla, pero cuando está hecho en algún lenguaje de alto nivel, será necesario realizar otros programas y añadirlos o bien modificar los ya existentes.

Conclusiones

Como se puede observar la metodología para el desarrollo de un sistema experto es fácil de asimilar. Sin embargo, el proceso de construcción de un sistema experto es lento y tardado. Las etapas que más tiempo llevan son la de Ingeniería del conocimiento y la de programación cuando se usa un lenguaje de alto nivel. Se dice que de los proyectos de software, el desarrollo de un sistema experto es de los más tardados.

Capítulo II

La Orientación Vocacional

En este capítulo se trata la parte teórica correspondiente a el dominio de nuestro sistema experto: La Orientación Vocacional. Cabe aclarar que no se pretende tener un tratado sobre Orientación Vocacional, tampoco discutir sobre las diferentes teorías al respecto, más bien el objetivo es tener una base teórica que permita el desarrollo del sistema experto.

El subcapítulo I.1, **Consideraciones Teóricas**, contiene una introducción al problema de la elección de carrera, describe las teorías clásicas de la Orientación Vocacional, enuncia los factores que influyen en el desarrollo vocacional del individuo y propone el plan de trabajo a seguir por un orientador vocacional.

Dado que nuestro sistema experto trabajará bajo la aplicación de tests, el subcapítulo I.2, **El Test Psicológico en la Orientación Vocacional**, contiene la teoría relacionada a esta herramienta. Define lo que es un test psicológico, los tipos de tests, como administrar un test y como puede implementarse un programa de tests en un proceso de Orientación Vocacional.

Por último, el subcapítulo I.3, **Sistemas de Archivo Escolar**, trata los sistemas de archivo escolar como una herramienta para el mejor conocimiento del individuo y como un medio para ahorrar tiempo en el conocimiento de sí mismo en el proceso de la Orientación Vocacional.

II.1

Consideraciones Teóricas

EL PROBLEMA DE LA ELECCION DE CARRERA

Una de las decisiones más importantes de toda persona es la elección de la actividad que desempeñará en su vida. La importancia de esta decisión radica en la influencia que tendrá la profesión que ejerza en su estilo de vida, su productividad e incluso en la realización de sus sueños. La actividad que desempeñe un individuo influye en la posición que ocupará en la sociedad, y en el conjunto de objetivos que persiga en su vida. Los sociólogos dicen que la influencia del empleo es tal que afecta nuestros valores, actitudes y hábitos y aún el tipo de personas con las que haremos amistad y nuestras identidades socioeconómicas.

De acuerdo con esto, el trabajo de un orientador vocacional consiste en brindar a los jóvenes el poder elegir una forma de vida en lugar de simplemente un modo de subsistencia. El y sus colegas tienen el deber de adoctrinar a los jóvenes en el conocimiento de sí mismos y en la toma de una decisión sensata y a tiempo. Dicha labor se torna difícil cuando pensamos en el hecho de proporcionar a un chico de quince años los medios necesarios para conocerse a sí mismo y plantear el curso de su vida en una etapa natural de inmadurez. No se trata solo de dar una simple presentación de opciones a elegir. De esta forma, a falta de condiciones ideales, el orientador tiene que echar mano de diversas herramientas.

LA NECESIDAD DE UNA TEORIA

Dentro del proceso de elección de carrera, existen ciertas limitantes. De acuerdo con la sociedad en que viva el individuo, éste tendrá la libertad de elegir

su profesión. Habrá sociedades muy primitivas que hagan que la libertad de elección sea nula, así como sociedades que cuenten con una amplia gama de profesiones y en donde la posibilidad de elección es amplia.

Todo proceso de orientación vocacional debe desarrollarse sobre la base de una teoría en donde se contemple su desarrollo, las principales variables, etc. De acuerdo con Ginzberg [1], existen tres enfoques generalizados en los que son empleados por los orientadores y sociólogos:

a) Teorías de casualidad.
b) Teorías de impulso.
c) Teorías de emparejamiento de talentos.

Teorías de casualidad

La experiencia de mucha gente hace exitosa la hipótesis de la casualidad. Por ejemplo, si preguntáramos a un carpintero la forma en la que eligió su oficio, podría decirnos que ocasionalmente encontró un letrero solicitando un aprendiz de carpintería y él lo tomó. Un escritor podría decirnos que eligió su profesión después de leer *El Quijote de la Mancha*. La hipótesis de la casualidad sugiere que la elección profesional de un sujeto está determinada por un estímulo externo que llega a ser trascendental. Sin embargo, aunque los estímulos externos llegan a ser determinantes, no podemos basar el proceso de la elección de profesión solo en ellos. Tan solo veamos que no todos los que leyeron el letrero de la solicitud terminaron como carpinteros y que muchos hemos leído *El Quijote* sin llegar a ser escritores jamás.

Aunque la teoría de la casualidad contempla un punto importante como son los factores externos, resulta incompleta en tanto no tome en cuenta los demás factores. En este sentido podemos señalar que el efecto que tengan los estímulos externos dependerá de la forma en que sean contemplados por el individuo.

Teorías de impulso

Esta teoría es todo lo contrario a la teoría de casualidad. Aquí se consideran únicamente los factores internos del individuo, se dice que la elección de profesión debe ser de acuerdo a las motivaciones internas, a veces inconscientes. Por ejemplo, un carnicero o un cirujano encuentran en su profesión una válvula de escape a sus impulsos sádicos, como también lo hace un pugilista o un torero; también podríamos hablar del homosexual que termina como peluquero.

[1] Ginzberg, E. y otros. "Occupational Choice, An Aproach to a General Theory", Columbia University Press, 1951.

Aunque en algunos puntos la hipótesis es acertada, sigue siendo incompleta al no considerar los factores externos. Por ejemplo, dentro de una misma profesión encontramos personas con caracteres muy variados o bien dos personas con diferente profesión tienen impulsos muy parecidos. Sin duda alguna, la elección profesional implica mucho más que solo encontrar un empleo que sirva como válvula de escape de la persona.

Teorías de emparejamiento de talentos

Estas teorías fundamentan su estudio en el llegar a una aproximación entre las aptitudes, los intereses y los valores del individuo con los patrones de alguna profesión. En muchos lugares, este enfoque es la guía para el proceso de orientación vocacional.

Algunos autores como Macrae[2] señalan que uno de los principales defectos de esta hipótesis es el tratar de terminar el proceso con una sola evaluación. Otros más han profundizado en este punto señalando que la orientación vocacional es un proceso evolutivo que ha de realizarse en varias etapas de la vida; la evaluación más confiable de entre varias, es la que está más cerca del final de la vida escolar del individuo.

Una teoría más completa

Ginzberg señala que la Teoría del Emparejamiento de Talentos sigue siendo incompleta ya que hacen falta elementos que aseguren una completa satisfacción en el desempeño de la profesión elegida desde este enfoque. Sin embargo, señala que una teoría completa sería una que se montase sobre este modelo parcial de emparejamiento de talentos. Su teoría contempla cuatro conceptos básicos:

1) La orientación vocacional es un proceso evolutivo que debe desarrollarse durante un período de años.

2) El proceso es mayormente irreversible.

3) El proceso involucra intereses, capacidades, valores y las oportunidades que se tengan.

[2] Macrae, A., "The Case for Vocational Guidance", Pitman, 1954.

4) Hay tres períodos en el proceso de la elección profesional:

a) Período previo de elección fantástica: se rige por el deseo de ser un adulto.

b) Período de elección de tanteo: está determinado por la aficiones y posteriormente se va modificando por las aptitudes y valores.

c) Período de elección realista: es el período en donde se llega a la conclusión del proceso eligiéndose la profesión.

Este modelo de Ginzberg fue desarrollado por Super[3] en una teoría que comprende diez puntos:

1. Las personas discrepan en aptitudes, intereses y personalidades. De acuerdo con esto, no existen dos personas que tengan características exactamente iguales.

2. De acuerdo a sus rasgos específicos, cada persona está capacitada para desempeñar un cierto número de ocupaciones. Las investigaciones han dejado ver que aún las personas impedidas física o mentalmente tienen potencial para desempeñar satisfactoriamente un gran número de profesiones.

3. Cada profesión tiene un patrón de aptitudes, aficiones y rasgos de personalidad con una cierta tolerancia que permite que una cierta diversidad de sujetos sean contenidos en una profesión. Es decir, así como un individuo puede desempeñar satisfactoriamente un número de profesiones, una profesión puede ser desempeñada satisfactoriamente por una amplia gama de sujetos.

4. El proceso de elección y adaptación es ininterrumpido debido a que las situaciones en las que los individuos viven y trabajan cambian. Por ejemplo, el proceso de elección y adaptación vocacional no concluye cuando el joven encuentra su primer empleo, ya que tendrá que pasar por una etapa en donde él se ajuste a aquellas situaciones no esperadas. Aún pasada esta adaptación, el joven va profundizando en su campo de trabajo y encontrando nuevas alternativas que le muevan a buscar lo que él considere un mejor trabajo.

5. El proceso de elección y adaptación vocacional se lleva a cabo en cuatro etapas de la vida: crecimiento, exploración, mantenimiento y declive. En la etapa de exploración se presentan las fases de fantasía, tanteo y realismo. En la de mantenimiento las fases de ensayo y estabilidad.

[3] Super, D., "A theory of vocational development", American Psychologist, 8, págs. 185-190, 1953.

6. El proceso se ve afectado por factores como el nivel socioeconómico, la aptitud mental, la personalidad del individuo y las oportunidades que se le brindan. Existen factores que determinan cómo el sujeto pasará por las diferentes etapas e influirán en el nivel profesional alcanzado, el número y duración de sus empleos de ensayo, etc.

7. El proceso de maduración de intereses y aptitudes del individuo puede ser orientado hacia un mejor aprovechamiento de tales. De acuerdo a varios estudios, las aptitudes y personalidad del individuo tienen raíz en la constitución neuronal y endocrina heredadas. Sin embargo, otras investigaciones han arrojado que se puede influir en el desarrollo de actividades del individuo de manera que sus aptitudes, intereses y rasgos de personalidad sean aprovechados.

8. El proceso de elección y adaptación vocacional puede concebirse en dos partes: primero, el conocimiento de sí mismo y segundo, la implantación de dicho concepto. El sujeto debe conocer sus cualidades y sus metas, visualizar lo que quiere ser y prepararse para ello, después llega al punto tal en el que enfrenta la situación real de lo que quiso ser y lograr un equilibrio entre lo ideal y la realidad. Según Super, éste es el meollo del problema en el proceso de elección y adaptación profesional.

9. El proceso de conjugar el conocimiento de sí mismo con la realidad es un juego de ensayo, es decir, se auxilia de situaciones tales como cursillos o clubes o bien trabajos de ensayo o estables en donde el individuo pueda confrontar sus expectativas y ser retroalimentado.

10. La satisfacción de desempeñar una profesión y la felicidad que se encuentre en la vida, dependen de hasta que punto la persona encuentre causes para el desarrollo de sus aptitudes, intereses, aficiones, rasgos de personalidad y valores. En otras palabras, dicha satisfacción estará en proporción a la distancia existente entre la imagen de vida que el individuo deseó tener y la que realmente tiene.

La teoría de Super está cimentada primeramente en el conocimiento de sí mismo y después en el conocimiento del medio profesional para así poder alcanzar una adaptación satisfactoria del individuo. En lo que continua de este texto, trataremos de emplear esta teoría junto con otros elementos para encontrar una secuencia de pasos para el trabajo vocacional.

FACTORES QUE INFLUYEN EN EL DESARROLLO VOCACIONAL

Cada individuo tiene un desarrollo vocacional de acuerdo a diferentes factores. De esta forma, el papel del orientador vocacional es ayudar a la persona a descubrirse a sí mismo, a descubrir el mundo profesional y poner las bases para una adaptación satisfactoria. Dentro de los factores que influyen para el desarrollo vocacional del individuo, resaltan dos instituciones: la familia y la escuela.

La familia

La familia es en la mayoría de los casos la base de formación del individuo. En un principio, los modelos vocacionales de la persona serán tomados exclusivamente de la familia. Algunos autores señalan un poco de inconveniencia en esto ya que al copiar modelos provenientes del padre o madre, habrá un error de al menos una generación de defasamiento. Esto es, el medio ambiente profesional cambia con el tiempo. Así, mientras generaciones atrás indicaban que el papel de la mujer era puramente doméstico, actualmente ésta desempeña una doble función hogar - empleo remunerativo.

La familia influye en aspectos tan visibles como lo es la religión. Diferentes estudios han mostrado que los valores religiosos asimilados influyen en la elección vocacional. Por ejemplo, es de prever que al tener una formación cristiana "protestante" en donde se resaltan la responsabilidad y la iniciativa personal, el individuo busque profesiones de carácter ejecutivo o hacia el autoempleo, mientras que no puede esperarse tanto esa actitud en una formación católico romana en donde se enfatiza la aceptación de la autoridad y adaptación conformista[4]. También podemos decir que la formación cristiana produce más servidores sociales y maestros que soldados, mientras que los musulmanes tienen preferencia por las carreras militares más que por el trabajo social.

De acuerdo con la forma que influye sobre el individuo, Carter[5] clasifica a las familias de la siguiente forma:

a) "El tipo de familias cuyas aspiraciones se concentran en el hogar": éstas están convencidas de la necesidad de planear con tiempo y su mayor aspiración es el triunfo de los hijos en la escuela y posteriormente en su profesión. Los padres apoyan totalmente el desarrollo académico de sus hijos impulsándolos incluso a que formen parte de grupos juveniles como los *boy scouts* o actividades religiosas como parte de la formación moral y social.

[4] Super, D., "The Psychology of Careers", Harper de Row, 1957.
[5] Carter M., Into Work, Penguin, 1966.

b) "El tipo de familia de clase trabajadora pura": en esta clase los padres parecen tener poco interés en la educación de sus hijos. No se enfatiza la necesidad de mejorar la posición social que se tiene ni se piensa en la necesidad de planear. Cuando los hijos llegan a la edad mínima posible para dejar la escuela, los padres no muestran interés porque ellos continúen estudiando. En pocas palabras, un niño dentro de este tipo de familia no recibe ningún impulso para desarrollarse en una profesión diferente a la de sus padres, hermanos o vecinos.

c) "El tipo de familia brutalmente desahuciado y marginado": son familias que viven al día y en donde no hay ningún interés por la futura vocación de los hijos.

Aspectos como el tamaño de la familia y la posición del sujeto respecto a los hermanos, influyen en el desarrollo vocacional del individuo. No es raro encontrar que un maestro sea una persona que fue la mayor de sus hermanos y a quien le fue encomendado su cuidado.

El grado de influencia que tenga la familia sobre el desarrollo vocacional del individuo varía. En muchos casos la opinión que dan los padres de familia es muchísimo más determinante que otros consejos que el individuo recibiera. En otros en donde la persona aspira a una profesión muy diferente a la de sus familiares, el mayor apoyo en su desarrollo vocacional y la mayor influencia será externa.

La escuela

Para muchos, la escuela es la segunda mayor influencia después de la familia. La manera en la que influye en el desarrollo vocacional es variada. Primero, es ahí donde el individuo palpa de una manera más clara sus áreas de interés y aptitudes. Usualmente, una buena instrucción en una rama específica del conocimiento, influye en que la persona elija tal o cual carrera. Como institución social, la persona se verá influenciada por las aspiraciones de sus compañeros y amigos de la escuela. En varios casos, el asesor o maestro es un modelo que es copiado por los alumnos para tener un lineamiento hacia una tendencia vocacional.

De manera general, los expertos han podido identificar los siguientes factores que influyen en la elección de carrera:

a) Factores personales: dentro de estos podemos identificar los intereses del individuo, sus aptitudes, habilidades y capacidades y en sí todo aquello que en su persona facilite o dificulte el desempeño de la carrera que elija.

b) Factores sociales y familiares: aquí nos referimos a aquellas influencias que se ejercen por parte de amigos, familiares, profesores y gente con la que el joven intercambia ideas.

c) Factores de expectativas de empleo y mercado de trabajo: aunque muchas veces el móvil principal para la elección de una carrera es el sentirse bien realizando cierta actividad, un factor importante es la remuneración y el campo de desarrollo que pueda tener el individuo. En este aspecto, existe gente que estaría de acuerdo en únicamente desempeñar su actividad sin importar cuanto dinero perciba, pero también aquellos que necesariamente vivirán de su carrera.

d) Características y demandas de las alternativas seleccionadas: en muchas ocasiones existe un tanto de limitación para estudiar una carrera dadas las escuelas que la imparten y las condiciones que se requieren durante el estudio. Aquí podemos mencionar factores como colegiaturas, tiempo, costo de la carrera, prácticas escolares, etc.

EL PLAN DE TRABAJO DEL ORIENTADOR VOCACIONAL

Si el trabajo del orientador vocacional está necesariamente vinculado con la futura vocación de sus clientes, sería obvio averiguar el significado de la palabra *vocación.*

Vocación: "se utiliza vocación para aludir a todo el esquema de vida de una persona, el cual incluirá factores laborales y extralaborales a la vez".

De acuerdo con los puntos hasta aquí desarrollados, se sugiere que el orientador vocacional organice su trabajo en cuatro partes:

1. Ayudar al individuo en el conocimiento de sí mismo.
2. Ayudar al individuo en el conocimiento del mundo laboral.
3. Ayudar al individuo a definir lo que desea de su trabajo.
4. Ayudar a definir lo que la persona espera de su vida extralaboral.

1. El conocimiento de sí mismo

El primer paso en el trabajo del orientador es ayudar al individuo a cristalizar un concepto claro sobre sí mismo. Nadie que no realice una meditación sobre lo que le hace diferente a los demás, como son sus valores, intereses, aptitudes, causas de satisfacción o de aflicción, podrá elegir su profesión de manera satisfactoria.

El conocimiento de sí mismo ayudará a la persona a fijar sus objetivos culturales, profesionales y existenciales de manera adecuada. Esta fase implica conocer:

a) Los intereses de la persona.
b) Sus aptitudes, es decir, lo que es capaz de hacer.
c) Sus valores y necesidades, es decir, lo que en general persigue en la vida.

2. El conocimiento del mundo laboral

Es importante dar a conocer de la manera más completa posible información sobre las profesiones que la persona tenga inquietud. Este trabajo es mejor mientras menos aspectos deje a la imaginación del individuo. Esto se logra con la debida documentación en folletos y libros informativos así como con la entrevista a los profesionales en activo.

3. El concepto profesional de sí mismo

En este punto el individuo desarrollará el concepto profesional de sí mismo, es decir, lo que espera de su profesión. El orientador ayudará a identificar las satisfacciones y beneficios que espera recibir de su trabajo.

De acuerdo con el Dr. Peter Daws[6], las satisfacciones en el trabajo pueden clasificarse de la siguiente manera:

a) Satisfacciones materiales: se contempla el ingreso que se tenga por desempeñar la profesión, la seguridad de conservar un empleo, la posibilidad de ascender, las prestaciones, etc.

[6] Investigación del Dr. Peter P. Daws sobre satisfacciones personales buscadas por los escolares, 1967.

b) El status: considera la posición que le de la profesión dentro de la sociedad, la importancia del puesto en la empresa así como el reconocimiento de los demás y la aceptación de sí mismo.

c) Pericia: es el desempeñar la profesión poniendo en práctica determinado conocimiento o habilidad. Es la existencia de la satisfacción por lo que se produce.

d) Valores: el desempeñar una profesión en donde se cubre la expectativa del valor que se quiere, por ejemplo de servicio social.

e) Sociabilidad: es el contacto que se tendrá con otras personas como son los compañeros de trabajo o con personas a quienes se les prestará un servicio.

f) Perceptivo: es el medio en el que se desempeña la profesión: viajes, al aire libre, etc.

4. El concepto extraprofesional de sí mismo

Aunque la profesión desempeñada influye grandemente en la vida del hombre, hay aspectos que quedan fuera de ésta. El aspecto extraprofesional es importante es y es donde el individuo responde a la pregunta de qué quiere ser en la vida. Podrá haber necesidades que sean cubiertas por el trabajo pero sin duda algunas no lo serán, en este sentido, el individuo deberá preocupase por el desarrollo integral de su persona. Esto me hace recordar a aquel niño al cual se le pregunta "¿qué quieres ser de grande?" y él responde "un hombre".

II.2

El Test Psicológico en la Orientación Vocacional

Qué es un test

Un test es básicamente un medio que permite examinar el comportamiento de una persona de tal manera que sea posible obtener deducciones generalizadas sobre su actuación y comportamiento. El test permite extraer una muestra de la persona para llegar a un calificativo de la misma. Es sólo un método entre varios, de reunir información sobre una persona.

Lo que en esta sección analizaremos es la utilidad de los test en el proceso de orientación vocacional.

De acuerdo con la teoría desarrollada en las secciones anteriores, la labor de la orientación vocacional se lleva a cabo cumpliendo con cuatro metas:

a) desarrollar el concepto de sí mismo del individuo;
b) desarrollar los conceptos profesionales;
c) desarrollar el concepto profesional de sí mismo;
d) desarrollar el concepto extraprofesional de sí mismo.

Los test pueden ayudar en varias etapas del proceso de orientación de la siguiente manera:

- Proporcionan información que ayuda al individuo a llegar al concepto de sí mismo.
- La información dada ayuda al asesor a sugerir líneas para la planeación de la vida de la persona.

Los test permiten obtener información sobre el proceso evolutivo en los intereses, aptitudes, personalidad y valores del individuo, dando pie a que de acuerdo sus metas, éste opte por desarrollar tal o cual cualidad. Como medio para obtener información, los test ayudan a desarrollar el concepto de sí mismo profesional y el concepto extraprofesional de sí mismo.

Los Bixler[7] señalan que cuando se apliquen test se debe tomar en cuenta los siguientes puntos:

⇒ Dar al estudiante datos estadísticos que respalden el resultado del test, por ejemplo: "50 de cada 100 chicos con su puntuación obtienen buenos resultados al ingresar a la carrera Lic. en Periodismo".

⇒ El asesor debe orientar al estudiante en la interpretación de los resultados, resaltando los puntos que él crea importantes.

⇒ Los resultados del test deben evaluarse de forma neutral.

⇒ El asesor debe aconsejar al estudiante en las decisiones importantes.

⇒ En caso que el estudiante dude, el asesor debe sugerir la eficacia del test y cuando sea necesario, ayudar al estudiante a encontrar las fallas.

⇒ Antes de la aplicación del test, el asesor debe conocer las expectativas del estudiante para aprovechar los resultados que posteriormente se obtengan y enfocar el proceso hacia donde convenga.

Otros puntos al aplicar un test son los siguientes:

- Debe usarse tests que vayan de acuerdo al nivel cultural y factores socioeconómicos del individuo.
- Los test pueden usarse en el proceso de toma de decisión del individuo y no sólo como una herramienta de predicción.
- Los resultados de un test deben ser enjuiciados con cuanta información se tenga disponible y comparados con otros medios.
- Los resultados de un test pueden variar de acuerdo a las circunstancias (cansancio, tristeza, hastío, etc.).

[7] Bixler, R.H.L. y Bixler, V.H., "Test interpretation in vocational counselling", Education and Psychological Measurement, 6, págs. 361-374, 1946.

- El propósito principal de los tests en el proceso vocacional es ayudar al estudiante en el conocimiento de sí mismo y de su concepto profesional.

- Los tests deben ser utilizados como una herramienta para eliminar algunas alternativas y considerar otras, haciendo más estrecho el camino de la elección vocacional.

- Antes de la implementación de los tests debe haber una etapa de orientación en donde se conozcan las expectativas del estudiante.

- Los tests dejan ver los puntos débiles y fuertes del individuo y pueden enseñarle que determinadas actitudes le ayudarían a corregir dichas debilidades.

- El estudiante deberá plantear sus propias conclusiones dados los resultados del test aún antes que el asesor sugiera.

- En ocasiones es conveniente no dar a conocer al estudiante propiamente los resultados del test, sino una interpretación de los mismos.

- El asesor debe tener conocimiento para poder hacer la selección, administración y a interpretación del test.

- Se debe procurar tener tantos datos como sea posible para validar los resultados del test.

Mientras que para algunos el test no es un medio confiable en el proceso de orientación vocacional, para otros se ha convertido en una herramienta poderosa. Como ya vimos, el test es sólo un medio en el proceso y cuando se aplica de manera adecuada, llega a ser de mucho provecho.

En la siguiente sección profundizamos más sobre que tipos de tests hay, cuando es conveniente utilizarlos y cómo utilizarlos.

TOPICOS SOBRE TESTS PSICOLOGICOS

Tipos de tests

Podemos clasificar los tests de dos formas:

a) De acuerdo a su función, es decir, de acuerdo a lo que pretenden medir.

b) De acuerdo con el procedimiento utilizado, esto implica presentación, diseño y naturaleza de las tareas a realizar.

Por su función, los tests se clasifican de la siguiente manera:

- Tests de capacidad general
- Tests de aptitudes
- Tests de consecución
- Tests de intereses
- Tests de personalidad
- Baterías en general

De acuerdo a su procedimiento, encontramos los siguientes tipos de tests:

- Velocidad versus potencia
- Individual versus grupo
- Objetivo versus subjetivo
- Ejecución versus papel - lápiz
- Normalizado versus confeccionado por el profesor

CLASIFICACION POR FUNCION

Tests de capacidad intelectual en general

Se conocen también como tests de aptitud escolástica, tests de capacidad mental o tests de inteligencia. Son los tests que más erróneamente se han usado. Uno de los motivos es que existen tantas definiciones de inteligencia que cada uno de ellos parece medir algo diferente. Son usados por diversas instituciones para el reclutamiento de gente.

Tests de consecución

Son empleados para medir el grado de avance en determinados aspectos. Cuando se emplean test normalizados, permiten la comparación de las actitudes de la persona con el patrón estándar.

Tests de aptitudes

Este tipo de tests son utilizados para estimar el futuro éxito de una persona en determinadas profesiones. Intentan medir aptitudes, entendiéndose por aptitud un conjunto de características del individuo que le permiten, mediante entrenamiento, adquirir algún conocimiento o técnica. Se trata de medir características supuestamente indispensables para el desempeño satisfactorio de una profesión.

La principal desventaja de este tipo de tests es que nunca se logra medir directamente una aptitud, sino que se hace a través de actuaciones, es decir, de algo que en determinado momento se aprendió. Esto no indica que no se deban usar los tests de aptitudes, ya que a final de cuentas representan una aproximación aceptable en la medición de tales características.

Tests de intereses

Este tipo de tests presentan al individuo un conjunto de entidades que le permitirán identificar sus intereses y aversiones. Usualmente, se le permite a la persona elegir entre varias actividades, profesiones o reacciones ante situaciones o personas. De acuerdo a sus respuestas, se le trata de ubicar en algún grupo profesional, es decir, emparejar sus intereses con los de un estándar de personas en determinadas profesiones.

Aunque el factor interés no es el que determina totalmente la elección de carrera, si es un aspecto importante a tomar en cuenta, considerando que se prevé mayor éxito cuando una persona disfruta lo que hace. Los investigadores sugieren que una vez pasada la edad de los 16-18 años, los intereses son estables, mientras que a temprana edad cambian rápidamente.

Hay que señalar que para efectos de orientación vocacional, no se puede deducir intereses únicamente de las aficiones o pasatiempos, por ejemplo, el que a una persona le guste correr no es necesariamente indicio de que se sienta bien trabajando al aire libre.

Antes que ocasionar un emparejamiento profesional, un test de intereses contribuye al análisis de sí mismo y a la motivación de un autoconocimiento.

Un problema que se presenta aquí es si se debe aceptar el perfil de intereses de la persona o si se le debe estimular al cambio. Un joven al que se le ha encaminado hacia una cultura basta, que ha tenido un buen acercamiento a la sociedad y que tenga un conocimiento de sí mismo relativamente bueno, podría beneficiarse plenamente de un consejo dado su perfil de interés. Por el contrario, un joven cuyas experiencias se reducen estrechamente a los familiar y escolar, podría cambiar radicalmente sus intereses al presentársele un medio ambiente más amplio con experiencias diferentes. Es aquí donde entran los términos direccional y adireccional de la orientación vocacional. El proceso de la orientación vocacional debe ser direccional en una etapa y adireccional en otra. Es direccional en tanto se le proporciona al estudiante medios para desarrollar el concepto de sí mismo, el concepto del mundo laboral y el concepto profesional y extraprofesional de sí mismo; entonces, cuando ya se le han dado todas las herramientas y bases, el proceso se torna adireccional dando libertad al estudiante para la elección profesional.

Tests de personalidad

El uso de estos tests cae en alguno de estos cuatro objetivos:

1. Obtener una autodescripción del joven.
2. Identificar problemas normales en el estudiante, que puedan ser solucionados eficazmente por el asesor.
3. Identificar problemas o desajustes serios para someterlos a tratamiento especial.

4. Proporcionar al estudiante información que le ayude al conocimiento de sí mismo y le mueva a reflexionar acerca de su adaptación a una profesión.

Estos tests se emplean del modo de emparejamiento de talentos, sin embargo este trabajo se torna más difícil que en el test de intereses, puesto que ciertas necesidades pueden satisfacerse en el desempeño no solo de una profesión, sino de varias. Sin embargo, el test de personalidad ayuda a que la persona obtenga un concepto general de sí mismo.

CLASIFICACION DE LOS TESTS POR SU PROCEDIMIENTO

a) *Ejecución versus papel y lápiz:* los tests de ejecución requieren que el individuo manipule objetos, por ejemplo, que arme un mecanismo, acomode cajitas, etc., mientras que el de papel y lápiz solo requiere que la persona escriba o marque sobre una hoja impresa. La ventaja de los tests de ejecución es que el asesor pueda observar movimientos y actitudes en el joven al realizar las actividades, lo cual en ocasiones es de más valor que los resultados cuantitativos. Por lo mismo, este tipo de tests suelen ser aplicados de forma individual.

b) *Individual versus grupo:* los tests individuales son aplicados por el experto calificado a un estudiante a la vez, observando así sus actitudes al ir respondiendo o ejecutando actividades. El test de grupo es aplicado a varias personas a la vez. No siempre es mejor aplicar un test de forma individual, hay tests que son elaborados específicamente para aplicar en grupo. Se usan tests individuales para evaluar estudiantes con dificultades y cuando se requiere observar cuidadosamente sus respuestas.

c) *Objetivo versus subjetivo:* Los tests objetivos son aquellos cuyas repuestas pueden ser evaluadas fácilmente, en dichos tests el estudiante marca una de varias posibles respuestas o escribe una palabra o frase dada una pregunta. Los tests subjetivos requieren de un proceso elaborado para la evaluación de respuestas. Los tests objetivos son de gran utilidad cuando se requiere identificar rápidamente puntos débiles o fuertes de una persona. Cuando los tests de interés, aptitud y personalidad son todos objetivos, apenas se obtiene un bosquejo de la persona. Los tests subjetivos son de utilidad cuando se considera el proceso de orientación vocacional como algo más que un simple proceso de emparejamiento entre personas y profesiones.

d) *Normalizado versus confeccionado por el profesor:* un test normalizado es aquel que se efectúa con instrucciones especiales y bajo ciertas normas. Se dispone de datos que permiten comparar los resultados obtenidos con los

arrojados por otros grupos de personas. Los tests confeccionados por el profesor, son los que realiza él mismo y le sirven como realimentación evaluando el progreso en el proceso. Ambos tipos de test son empleados en el proceso de orientación vocacional.

e) *Velocidad versus potencia:* los tests de velocidad son aquellos en los que el estudiante debe contestar tantas preguntas como le sea posible en un determinado período de tiempo. Si en estos tests no hubiera límite de tiempo, el estudiante posiblemente podría contestar todos los conceptos del test. Aquí se evalúa la velocidad para formular una respuesta. En los tests de potencia se evalúa la amplitud y profundidad en los conocimientos de una persona, eliminando hasta donde sea posible el factor tiempo. En los tests normalizados, existe un límite en el tiempo dado el promedio que tardan en contestar las personas, pero en sí el tiempo no es una presión para contestar este tipo de tests. Los tests de potencia ordenan sus reactivos por dificultad, es decir de lo más fácil a lo más difícil. Los tests de velocidad no suelen emplearse para evaluar intereses ni personalidad. En ellos, las personas que tienen dificultades son aquellas que con extremada prudencia tienen que checar una y otra vez sus respuestas, los que tardan en captar un tema o aquellos con problemas físicos como deficiencia visual o problemas de artritis. Cuando se requiere medir capacidad, generalmente se emplean test de potencia, excepto cuando un empleo requiere que se desempeñe con cierta rapidez.

COMO ADMINISTRAR UN TEST

Se recomienda que el test se aplique en cuartos bien iluminados, con muebles adecuados para la escritura y con un mínimo de ruido y otras distracciones. Se dice que es mejor aplicarlo entre las 10 de la mañana y las 3 de la tarde, ya que la persona está más dispuesta en este tiempo.

Con respecto a la forma de aplicarlos, existen dos enfoques:

a) *Test de saturación:* de acuerdo a este enfoque, el asesor selecciona todo el material que considera necesario para el proceso de orientación vocacional y lo aplica a un grupo de estudiantes. El supuesto que respalda este enfoque es que la mayoría de las aptitudes e intereses son importantes y que es mejor evaluarlas al mismo tiempo para después presentar los resultados al estudiante y discutirlo. Las ventajas de este enfoque son el ahorro de tiempo y de recursos. Las desventajas son la escasa participación del estudiante y la poca motivación que se le da para el proceso.

b) *Test de precisión:* en este enfoque ambos, estudiante y asesor, identifican los puntos a tratar y la aplicación de un test se da en el curso del proceso. Una vez hecho, se evalúa y se discuten los resultados, entonces pueden o no planear la administración de otro test. Las ventajas son que existe una verdadera interacción entre el asesor y el estudiante, hay motivación y hay participación por parte de este último. Los resultados de los tests ayudan para la comunicación entre ambos, en lugar de ser una barrera. Las desventajas se centran en el empleo de tiempo y recursos.

Aunque el enfoque de test de precisión es el ideal que debiera perseguir todo instituto, por motivos de tiempo y recursos, en ocasiones será necesario aplicar tests de grupo. Al hacerlo se recomienda tomar en cuenta los siguientes puntos:

- Esforzarse por que la aplicación de los tests no se haga de manera consecutiva uno tras otro, esto causaría cansancio y falta de motivación en los estudiantes.
- Cuando los tests se administren a grupos numerosos, evitar cualquier posible interrupción, hacer que los estudiantes se sienten en sillas alternas, fijar un horario y fomentar un ambiente amistoso entre el asesor y los estudiantes.
- Tener preparado todo el material indispensable para la aplicación de los tests.
- Procurar aplicarlos en un tiempo adecuado como lo sería el final de año en donde se tiene próxima una toma de decisión para los estudios.

IMPLANTACION DE UN PROGRAMA DE TESTS

El implantar un programa de tests que ayuden a la orientación vocacional es un proceso que debe llevarse a cabo cuidadosamente. Para hacerlo se sugiere primero pensar en lo siguiente:

a) Qué tests serán incluidos.

b) Para qué edad son apropiados.

c) Hacer una breve discusión sobre el posible mal uso de dichos tests.

d) Solicitar la siguiente información sobre cada test :
- Los propósitos del test y de que manera pueden emplearse los resultados.
- Cómo administrar el test, puntuarlo e interpretar los resultados.
- Los datos de normalización (si es un test normalizado).
- Datos sobre su fiabilidad.

El lugar del test en el proceso de Orientación Vocacional

Hay que recordar que el test es solo una parte del proceso de orientación vocacional, existen otros métodos como la observación y las entrevistas. Algunos autores señalan como ventajas de los tests las siguientes: ahorran tiempo, son objetivos, son fiables. Algunas de sus limitaciones son: a veces son difíciles de usar y difíciles de interpretar, revelan aspectos limitados del sujeto y son inflexibles.

Con la implantación de un programa de tests no se elimina el uso de los demás recursos sino que al complementarse el proceso se hace más completo.

¿Qué tests elegir y cómo obtenerlos?

Tal vez sea ésta la etapa más difícil en la implantación de un programa de tests. Aquí se debe responder a las siguientes preguntas:

- ¿Dónde buscar los tests?
- ¿Qué tests se adaptan mejor a la finalidad que se tiene?
- ¿Qué tests se está capacitado a implementar?

Respecto a la primera pregunta, a continuación se muestran algunas editoriales que distribuyen tests:

TEA, Fray Bernardino de Sahagún, s/n, Madrid 16.
SURGAM, Padre Amigó, 5, Madrid 19.
INAPP, Paseo de la Habana, 66, Madrid, 16.
INSTITUTO PONTIFICIO SAN PIO X, Tejares, Salamanca.
MEPSA, Francos Rodríguez, 47, Madrid 20.
HERDER, Balmes, 26, Barcelona 7.
PAIDOS, KAPELUSZ, ESPASA-CALPE (distribuye Herder).
SECADAS, J.S., Marcos Robles, Simancas, 7, H, 1o. Dcha, Madrid, 29.
CPA, 48 Avenue Victor Hugo, 75783 Paris Cedex, 16.
EAP, 6 rue André-Chenier, 92 Issy les Moulineaux.
Martínez de Murguía, Valdeverde, 30, Madrid, 13.
KELVIN (material), Miguel Yuste, 43, Madrid, 17.

Algunos puntos a tomar en cuenta en la selección de tests son los siguientes:

1. Uno de los problemas que se dan en la aplicación de tests, es cuando se trata de adaptar un test desarrollado para una sociedad diferente. En este aspecto, habría que buscar tests normalizados sobre una población lo más parecida posible a la propia geográficamente hablando y tomando en cuenta los factores social, sexo y edad.

2. Elegir tests que puedan ser evaluados de manera objetiva, rápida y económicamente. La rapidez es importante para no interrumpir el proceso de orientación, sino al contrario, al evaluar el test tener en poco tiempo más información que permita continuar.

3. Usar versiones de tests que estén debidamente actualizadas.

4. Evaluar el test en cuanto a la fiabilidad y validez que haya tenido. Aquí habría que examinar en qué condiciones el test ha dado buenos resultados y compararlos con la situación que se tiene.

Es importante señalar que para no caer en el abuso en el uso de tests, la gente que los utilice debe tener cierta preparación. Para empezar, debe tener el conocimiento de que se debe seguir detenidamente las instrucciones del manual del test. Debe saber también que es necesario brindarle a la persona un ambiente amigable para tranquilizarlo y quitar cualquier confusión que éste tenga acerca de la aplicación del test. Cuando la persona no tiene conocimiento, podría pensar que los resultados del test son definitivos y hay que recordar que a final de cuentas lo que obtenemos del test es sólo un diagnóstico. De esta manera podemos ver que quien aplica un test debe saber mucho más que solo repartir hojas y lápiz o manejar un cronómetro.

Al aplicar un programa de tests, debemos tener cuidado en los siguientes puntos:

- No implementar un programa de tests simplemente por seguir la moda, el hacerlo requiere tener todos los recursos para el buen funcionamiento del programa.
- Tener bien definido el propósito que se persigue para el uso de test.
- No pensar que por estar aplicando un amplio número de tests el programa de orientación es bueno. Hay que recordar que el proceso de orientación se auxilia de varias herramientas y que la realización de entrevistas y las sesiones de consejo, aunque más tardadas que los tests, son parte importante del mismo.

CONTENIDO DE UN PROGRAMA DE TEST

Ahora surge una pregunta evidente, ¿qué tests debería contener un programa de tests? Aún es difícil describir un programa de tests ideal, sin embargo, podríamos decir que un programa de tests debería contener al menos:

1. Medidas de consecución de objetivos en materias específicas.
2. Un test general de aptitud.
3. Una medida de ajuste personal y social.
4. Una medida de intereses.

Medidas de consecución de objetivos

Aquí se utilizan tests diseñados por el profesor para medir el progreso del estudiante en sus materias. De esta forma se lleva un registro acumulativo y se observan las tendencias de aprovechamiento del joven, lo cual proporciona un buen parámetro para la toma de decisión. En el *cuadro 2.1* se muestra un ejemplo.

Tema	Año 1990 %	1991 %	1992 %	1993 %
Idioma Inglés	60	71	50	65
Matemáticas	54	43	30	40
Física	61	60	50	45
Biología	65	70	63	61
Geografía	52	55	61	57
Historia	69	63	78	82
Español	65	45	58	51

Cuadro 2.1. *Ejemplo de un registro acumulativo de un test de consecución.*

Test de capacidad general

Si en el test de consecución comprobamos el avance del joven en sus materias, el test general de aptitud muestra ahora la capacidad del mismo en las diferentes ramas del conocimiento. El caso ideal es cuando los resultados de ambos tests concuerdan, por ejemplo cuando el test de consecución arroja resultados de progreso en matemáticas y el de capacidad general muestra una puntuación alta en esta área. Si los resultados discrepan, tenemos dos posibles casos:

a) *Subconsecución:* es cuando el desempeño académico de la persona está por debajo de la puntuación obtenida en el test de capacidad general. Aquí podríamos pensar que o bien el estudiante requiere dar mayor esfuerzo en sus estudios o bien que la estructura académica de la escuela está fallando.

b) *Superconsecución:* para los investigadores este caso es descabellado, es cuando la persona tiene un desenvolvimiento académico mejor a la puntuación de inteligencia obtenida, e indican que esto es una evidencia de que no se pueden dar explicaciones automáticas de la discrepancia entre los dos aspectos: consecución medida e inteligencia medida.

Tests de personalidad y ajuste social, e interés

Hay que recordar que los factores personalidad, ajuste social e interés son importantes en la toma de decisión. Aunque un test está limitado en cuanto a dar una descripción total del comportamiento humano por factores tales como la motivación o el estado de ánimo en el tiempo de la aplicación, proporciona una buena aproximación cuando se usa de manera adecuada.

Conclusión

Hoy en día la mayoría de los tests están bien proyectados y diseñados, por lo que siempre y cuando sean usados acertadamente y cuando el programa de tests sea completo, se contribuye a reducir considerablemente el campo de elección en el proceso de orientación vocacional.

II.3

Sistemas de Archivo Escolar

En un tiempo como hoy y en un país como el nuestro, la mayoría de los estudiantes parecen pasar por la escuela sin ser totalmente advertidos. En una escuela, un sistema de archivo escolar contribuye a dar una atención especial a cada alumno, situación que hoy en día parece muy lejana. Un sistema de archivo debiera contener información que ayude al proceso educativo, de esta forma se cumple el ideal de "educación para toda clase de personas, ofreciendo diversidad de instrucción y capacitación como fuera conveniente, de acuerdo a la edad, capacidad y aptitudes". También un sistema de archivo permite hacer diagnósticos en el caso de alumnos rezagados y con problema de comportamiento.

En el proceso de orientación vocacional, un sistema de archivo puede ayudar en los siguientes sentidos:

- De acuerdo con la formación cultural del individuo, dirigir el proceso de orientación.
- De acuerdo a la información académica, el orientador podrá hacer preferencias para dar información de ciertas profesiones.
- Reduce el tiempo de conocimiento del individuo teniendo ya información disponible.

Tipos de registro

Existen dos formas de manejar un sistema de archivo: registro acumulativo y registro anecdótico.

Registro acumulativo

Se lleva un registro de información acerca de la formación cultural del estudiante. Un buen registro acumulativo cumple con las siguientes características:

1. Es unitario, es decir, toda la información de una persona está concentrada y no en varios informes.

2. Es explícito y en su caso se proporcionan instrucciones acerca de cómo interpretar la información.

3. En su elaboración deberían participar todos aquellos que tengan que trabajar con él: orientador, profesores, etc.

4. Mostrar información a partir de la educación primaria.

5. El acceso a la información es sencillo y rápido.

6. Carece de comentarios enjuiciadores y sólo contiene información objetiva.

El *cuadro* 2.2 muestra un ejemplo de un registro acumulativo.

1. Datos de identificación: esta información es indispensable para el conocimiento general de la persona, para trámites oficiales es importante tener datos exactos.

2. Información de la salud: contiene datos sobre posibles complicaciones en la salud del individuo, la justificación de tener información de este tipo está en localizar las limitantes que pudiera tener la persona debido a su salud. Además de considerar deficiencias físicas que impiden el buen desempeño en una profesión. puede haber limitantes de tiempo si por ejemplo, la persona asiste periódicamente a un tratamiento.

***Cuadro 2. 2** Contenido de un archivo acumulativo.*

1. Datos de identificación:	*Nombre completo.* *Sexo.* *Fecha de nacimiento.* *Fotografía.*
2. Información de la salud:	*Asistencia a clínicas y hospitales.* *Verificar lista de enfermedades.* *Estado general de salud.* *Defectos físicos.* *Fechas y resultados de las revisiones médicas.*
3. Hogar e historial familiar	*Dirección y número de teléfono.* *Nombres completos de padres o tutores.* *Divorciados, separados, fallecimos.* *Profesión del padre y lugar de trabajo.* *Profesión de la madre y lugar de trabajo.* *Orden de nacimiento.* *Religión.* *Subsidios (comidas, uniformes, etc.).* *Involucración de los padres con la escuela.*
4. Historial escolar:	*Nombres de escuelas anteriores.* *Fechas de admisión y de partida.* *Tasas de asistencia (razones de las ausencias).* *Materias estudiadas y grados obtenidos.* *Datos del traslado y motivos de los mismos.*
5. Resultados de tests normalizados:	*Nombre del test.* *Fecha de administración.* *Puntuación y significado.*
6. Datos de personalidad:	*Verificar lista de conceptos sobre actitudes hacia sí mismo.* *Actitud hacia otras personas, actitud hacia el trabajo.*
7. Orientación vocacional:	*Inclinaciones del estudiante.* *Inclinaciones de los padres.* *Recomendaciones de los orientadores vocacionales.* *Acción convenida sobre ulterior educación / preparación / empleo.* *Empleo a tiempo parcial.* *Notas sobre vocación postescolar.*

3. Hogar e historia familiar: aquí se contemplan todos los datos referentes a lo familiar y el hogar, que puede ser de gran utilidad en el proceso de orientación. Algunos datos no muy comunes son de importancia, como el hecho de que si los padres se han divorciado o alguno ha fallecido, asimismo el lugar que ocupa el muchacho respecto a sus hermanos. Información como la religión permite prever algunas tendencias del joven. En ocasiones es preferible que la información se obtenga en la forma de elegir una opción en vez de llenar un espacio en blanco por razones de objetividad. En el caso de la situación conyugal de los padres, bien se podría tener una presentación como la siguiente:

Vive con: padres / madre / padre / custodio o tutores / otro (especificar)

Detalles: divorcio / separación / fallecimiento

4. *Historial escolar:* aquí se dispone de información útil para motivos académicos. Los datos de asistencia son de provecho y se recomienda que sea de la forma de subrayar opciones, por ejemplo:

Ausente con frecuencia por un día o medio día / tiene largas ausencias / los padres justifican las ausencias / no vuelve por ayudar a sus padres.

Enseguida se puede disponer de un espacio en blanco para especificar los motivos cuando se requiera.

5. Resultados de tests normalizados: aquí es importante que se registren los resultados y la interpretación de los mismos, así como la fecha en la que se aplicaron los tests. Con esta información se ve el progreso de la persona en su paso por la escuela y se pueden hacer comparaciones con otras.

6. Datos de la personalidad: en términos de Psicología, personalidad es el resultado de la interacción del temperamento del individuo (determinado genéticamente) con su medio ambiente. Para no caer en comentarios subjetivos y enjuiciadores, la información debe capturarse en forma de subrayado de opciones, por ejemplo:

Cuando contesta a preguntas: dispuesto siempre a contestar / a veces impaciente / a veces no hace caso / impaciente salvo cuando está de mal talante / se pone nervioso, se ruboriza, da voces al contestar / no tímido pero sí apático.

Actitud de otras personas: gusta / no gusta / indiferente / suele confraternizar con tipos inestables / le hacen trampas, le engañan.

Se dice que la información que se debe contemplar es de tres tipos:

a) Relaciones de un estudiante con otras personas.
b) Relaciones de un estudiante con los alumnos.
c) Actitud de un estudiante hacia el trabajo escolar y actividades fuera de la escuela.

7. Orientación vocacional: aquí se concentra toda la información que de manera directa puede influir en las preferencias vocacionales de la persona. Información como los empleos a tiempo parcial pueden situar puntos de partida para el orientador.

Registros anecdóticos

Existen hechos que por sí solos no pueden ser anexados a un registros acumulativo. Tal vez la conducta observada durante una clase no sea un dato importante, pero si registramos la conducta presentada durante todo el año si podremos deducir información confiable e importante.

Los registros anecdóticos son manejados por la gente que está en constante contacto con los estudiantes, como son los profesores, y son hechos de tal manera que puedan ser usados a diario. Deben ser objetivos, de manera que no se permite escribir en ellos comentarios sino hechos. Por ejemplo, en lugar de escribir "Pedrito es muy grosero conmigo", registrar "hoy Pedrito me llamó elefante ridículo".

Conclusión

La importancia de un sistema de archivo en el proceso de orientación vocacional, radica en permitir conocer aspectos importantes de la persona ahorrando tiempo en el conocimiento del mismo.

Capítulo III

Desarrollo del Sistema Experto

Una vez tratadas la Teoría de los sistemas expertos y la Teoría sobre Orientación Vocacional, podemos comenzar la descripción del desarrollo de nuestro sistema experto.

El presente capítulo es la descripción del proceso de desarrollo de nuestro *Sistema Experto en Orientación Vocacional y Guía de Carreras en el Estado de México,* siguiendo con la metodología vista en el capítulo I. En cada uno de los subcapítulos se describe una etapa del proceso. De esta forma, tenemos los siguientes subcapítulos:

III.1. Definición del problema.
III.2. Consideración de las alternativas.
III.3 Problemas para sistemas expertos.
III.4. Estimación de las inversiones y beneficios.
III.5. Elección de una herramienta de desarrollo.
III.6. Aplicación de la Ingeniería del Conocimiento.
III.7. Desarrollo de la base de conocimiento.
III.8. Desarrollo del software.
III.9. Comprobación y validación del sistema.
III.10. Mantenimiento del sistema.
III.11 Conclusiones.

III.1

Definición del Problema

El problema de la elección de carrera de un joven de bachillerato fue tratado en el capítulo II. Se alcanza a entender entonces que es necesario que todo estudiante de preparatoria reciba atención por parte de un orientador vocacional. Mencionamos que éste debe, entre otras cosas, ayudar al joven en el conocimiento de sí mismo, el conocimiento del mundo laboral y el conocimiento profesional y extraprofesional de sí mismo. Pudimos observar también que una de las herramientas empleadas por el orientador vocacional son los tests psicológicos.

Con estos antecedentes y tomando en cuenta que la educación pública en nuestro país no permite menos de 30 alumnos por orientador vocacional, el proceso de orientación se torna pesado al tratar de atender a cada persona por separado. Aunque siempre es necesario el contacto personal con el estudiante, si pudiéramos idear un mecanismo por medio del cual la aplicación de test fuera más rápida y confiable, permitiendo obtener resultados de los mismos en un período de tiempo menor.

III.2

Consideración de las Alternativas

Una de las opciones consiste en buscar gente capacitada para la aplicación de tests y para la evaluación de los mismos; sin embargo, en universidades públicas esto no es posible debido al restringido presupuesto.

El uso de manuales de autoorientación podría ser una solución, sin embargo es una solución tardada al tener que imprimir dichos manuales y además siempre se requiere la supervisión directa de un experto. Podría también resultar un poco costoso y en algunos casos inaccesible para el estudiante.

No existe una aplicación de software convencional para la solución del problema por lo que podríamos pensar en un sistema experto.

III.3

Problemas para Sistemas Expertos

El nuestro es un problema de diagnóstico y enseñanza por lo que, de acuerdo al capítulo I, puede ser resuelto por un sistema experto. Además se cumplen las siguientes características:

a) Necesidad de especialistas con conocimientos adecuados: resulta claro que un joven no puede ser orientado vocacionalmente por cualquier persona sino por una persona capacitada y con experiencia en el área.

b) Disponibilidad de especialistas: para la elaboración de nuestro sistema experto, contamos con la colaboración los orientadores del Departamento de Orientación Vocacional de la Escuela Nacional Preparatoria no. 7, quienes si bien no se dedicarán exclusivamente a este proyecto, si podrán atendernos en lo que sea necesario.

c) Un campo limitado: el problema está perfectamente delimitado a una herramienta para orientación vocacional de jóvenes que estudian bachillerato.

d) Naturaleza del conocimiento: el conocimiento a emplear se puede representar simbólicamente.

e) Se tiene una alta posibilidad de recuperar las inversiones si contamos con que los beneficiados serán cientos de personas por año y que el sistema experto ofrece solución durante un período considerable de tiempo.

f) El sistema experto no necesitará de sentido común para resolver el problema.

g) Las soluciones que podemos encontrar no son ni muy complejas ni muy sencillas, por lo que es sistema experto es una buena alternativa.

h) Las soluciones a nuestro problema son encontradas por medio del razonamiento.

i) El sistema experto analizará un número no mayor a las 500 carreras, las cuales representan una solución para el sistema. Considerando la capacidad de procesamiento de las actuales computadoras personales, el sistema experto sigue siendo una buena alternativa.

III.4

Estimación de Inversiones y Beneficios

Primeramente estimemos los costos que traerá implementar un sistema experto en una institución pública. Empezando por el equipo necesario para la implementación del sistema experto, diremos que en la mayoría de las instituciones de educación media básica se cuenta con centros de cómputo en donde el sistema podrá ejecutarse. Si pensamos específicamente en el sistema de educación media de la Universidad Nacional Autónoma de México, cada plantel cuenta con la infraestructura de cómputo requerida por el sistema, por lo que esto no implica gasto alguno. Por lo que a software se refiere, se cuenta con los compiladores necesarios para el desarrollo.

Pasando a los honorarios de los especialistas, diremos que cada plantel cuenta con un departamento de orientación y que las personas que ahí laboran ya reciben un salario por lo que su cooperación con el proyecto no implica gasto alguno. En el caso específico de el sistema experto desarrollado en esta tesis, se contó con la ayuda de expertos que ya laboraban en la institución. Por otra parte, el personal experto en computación estuvo formado por un servidor y dado que este es un trabajo de tesis, no hubo necesidad ni forma de cobrarle honorarios a nadie.

En conclusión, implementar nuestro sistema experto prácticamente no implica hacer ningún gasto.

Pasando a los beneficios, difícilmente se puede estimar una cantidad de dinero. Sin embargo, podemos notar una mejor atención por parte del departamento de orientación a los alumnos y la posibilidad de ahorrar tiempo y esfuerzo en la aplicación de tests. También se libera de una rutina al orientador al quitarle la responsabilidad de aplicar y evaluar los tests.

Visto de esta manera, parece que no invertimos en casi nada y los beneficios pueden ser considerables.

III.5

Elección de una Herramienta de Desarrollo

Las alternativas en este aspecto son principalmente tres:

a) Utilizar un lenguaje de programación.
b) Utilizar un lenguaje de Inteligencia Artificial.
c) Utilizar un shell de inducción.

Se eligió un lenguaje de programación y particularmente Turbo C++ (el compilador C++ de Borland) por las siguientes razones:

- Se tiene experiencia en dicho lenguaje.
- Permite hacer una interface con el usuario muy amigable.
- Permite la representación del conocimiento.
- Permite hacer el proceso de inferencia de manera optima.
- Se cuenta con el compilador.

III.6

Aplicación de la Ingeniería del Conocimiento

El conocimiento necesitado para el sistema experto fue recopilado de diferentes maneras. Se tuvieron entrevistas con las personas del Departamento de Orientación Vocacional de la Escuela Nacional Preparatoria plantel "Ezequiel A. Chávez" (7) especialmente con el M. Héctor Magaña. Fue ahí donde se obtuvo bibliografía la cual fue leída así como algunos tests psicológicos y guías de carreras. También se visitó dos veces la exposición de Orientación Vocacional organizada por la U.N.A.M. en la explanada del museo Universum en donde se pudo observar el funcionamiento de un Sistema Experto en Orientación Vocacional, utilizado actualmente por la Dirección General de Orientación Vocacional de la U.N.A.M.

Después de haber leído y haber consultado a nuestros expertos, se eligió como base de nuestro conocimiento un manual de autoorientación editado por la Dirección General de Orientación Vocacional[1].

Se eligieron 5 cuestionarios que permiten llegar al conocimiento del individuo, al conocimiento del mundo laboral y al conocimiento profesional y extraprofesional del individuo, dichos cuestionarios son los siguientes:

1. Cuestionario de Interés de Herrera y Montes.
2. Cuestionario de Habilidades de Herrera y Montes.
3. Tabla de valores.
4. Catalogo de ocupaciones, extraído del Catalogo Nacional de Ocupaciones.
5. Areas de estudio profesional.

[1] "Un mundo por conocer" Manual de autoorientación vocacional para alumnos de bachillerato. González Girón Gilberto. Dirección General de Orientación Vocacional, Secretaría de Servicios Académicos. U.N.A.M., 1992.

El primero permite identificar los intereses del individuo, el segundo permite conocer sus habilidades, la tabla de valores nos muestra que aspectos son importante para el individuo y que pueden influir en el curso de su vida. Posteriormente se muestra al individuo las ocupaciones existentes y se le pregunta cual de ellas le interesan (conocimiento del mundo profesional) y por último, en las áreas de estudio profesional, se identifica las áreas de estudio más adecuadas para la persona de acuerdo a sus antecedentes académicos.

El Apéndice B muestra describe detalladamente cada cuestionario, mostrando sus preguntas y como son evaluados.

En su totalidad, los cinco cuestionarios tienen como fin aproximar a la persona a aquella carrera cuyo perfil se acerque más. Cada carrera tiene un perfil ideal compuesto por una personalidad afín, una rama y campo ocupacional y un área de estudio. Las respuestas de los cuestionario nos permiten obtener estos tres factores para el individuo. El proceso es como sigue:

1. El cuestionario de intereses nos permite obtener un perfil de interés, el manual indica diez áreas de interés que son las siguientes:

a) Asistencial.
b) Ejecutivo-persuasivo.
c) Verbal.
d) Artístico-plástico.
e) Musical.
f) Organización.
g) Científico.
h) Cálculo.
i) Mecánico-constructivo.
j) Actividad al aire libre.

En el apéndice B se describen cada uno de estos tipos de interés.

El cuestionario permite obtener una puntuación para cada área de interés, de las cuales se seleccionan las tres más altas, ese es nuestro primer resultado.

2. El cuestionario de habilidades permite obtener una puntuación para cada una de las siguientes diez áreas de habilidad:

a) Servicio social.
b) Ejecutivo-persuasivo.
c) Verbal.

d) Artístico-plástica.
e) Musical.
f) Organización.
g) Científica.
h) Cálculo.
i) Mecánica-constructiva.
j) Destreza manual.

En el apéndice B se definen estas áreas de habilidad.

De aquí se seleccionan las tres áreas de habilidad con una puntuación más alta.

3. El siguiente paso es seleccionar las tres mejores parejas interés-habilidad, es decir, seleccionar aquellas áreas que presentan conjuntamente las mejores combinaciones de intereses y habilidades. El criterio para seleccionar estas parejas es el siguiente:

a) El mejor interés que es a la vez la mejor habilidad. Este es un caso ideal ya que tanto hay interés en el área como se tienen la habilidad para desempeñarla.

b) Si no se presenta el caso anterior, seleccionar el área que ocupa el segundo lugar en interés y el primero en habilidad.

c) Si no se presenta ninguno de los dos casos anteriores, seleccionar aquella área intermedia, por ejemplo, la que sea el segundo mejor interés y la segunda mejor habilidad.

d) Finalmente, seleccionar las tres mejores combinaciones de interés-habilidad, entre las que debe estar el primero y segundo lugar en interés y el primero en habilidad.

4. Enseguida se aplica el cuestionario de valores, el cual nos permitirá identificar que valores son importantes para el individuo.

5. Con las tres mejores parejas interés-habilidad (perfil de interés-habilidad) y los valores, podemos obtener un tipo de personalidad. El manual define seis tipos de personalidad, cada una con ciertos intereses y habilidades y valores (ver apéndice B). Dichos tipos de personalidad son los siguientes:

a) Realista.
b) Investigativo.

c) Artístico.
d) Social.
e) Emprendedor.
f) Convencional.

De esta forma, se seleccionan los tres tipos de personalidad a los que mejor se acerca el individuo. Con esto ya tenemos un primer factor a tomar en cuenta para la elección de carrera: la personalidad.

6. Se aplica el cuestionario de ocupaciones en donde el individuo señalará aquellas que le son de interés.

7. Se aplica el cuestionario de áreas de estudio profesional, en donde se obtendrán aquellas que son adecuadas para la persona de acuerdo con sus antecedentes académicos.

8. En este punto ya se tienen los tres factores a tomar en cuenta para la elección de carrera: personalidad, rama y campo ocupacional y área de estudio. El apéndice C muestra el perfil requerido para cada carrera, por lo que ahora habrá que tomar en cuenta aquellas carreras a las cuales más se acerca el individuo.

III.7

Desarrollo de la Base de Conocimiento

Definición de las soluciones

Como se mencionó en el capítulo I, el primer paso para el desarrollo de la base de conocimiento es la *definición de soluciones.* De acuerdo con el método de orientación elegido, el primer resultado que hemos de dar es un tipo de personalidad dado ciertos intereses, habilidades y valores. De esta forma, tenemos seis soluciones que son los tipos de personalidad:

a) Realista
b) Investigativo
c) Artístico
d) Social
e) Emprendedor
f) Convencional

Posteriormente, junto con la personalidad, debemos recabar aquellos campos y ramas ocupacionales que son del interés de la persona, así como sus áreas de estudio. Estos tres factores nos permiten llegar a nuestra solución final, que son aquellas carreras que se acercan más al perfil del individuo. Tenemos un total de 449 carreras, o dicho de otra forma 449 posibles soluciones, las cuales se describen en el apéndice C.

Definición de los datos que hay que suministrar al sistema

Para que nuestro sistema experto pueda llegar a la solución, necesitamos la que el usuario responda a cada uno de los cinco cuestionarios descritos en el apéndice B, los cuales son:

a) Cuestionario de intereses
b) Cuestionario de habilidades
c) Cuestionario de valores
d) Catálogo de ocupaciones
e) Areas de estudio profesional

Además, aunque no es determinante para la solución del sistema, se requiere de los datos personales del usuario, los cuales son: nombre, fecha de nacimiento, edad, sexo, estado civil, escolaridad, ocupación, domicilio y teléfono.

Dado que la cantidad de soluciones no es muy grande, no consideramos necesario una *jerarquía* ni un *árbol de decisión.* Nuestra idea es decir aquellas cinco carreras que mejor se acercan al perfil del usuario y mostrar en que grado las demás son viables, esto dando una explicación del porque.

Preparación de una matriz

Como se vio anteriormente, una de las formas de organizar el conocimiento es a partir de una estructura de matriz. Cuando se usa como herramienta de desarrollo un shell de inducción, la base de conocimiento será en sí la estructura de matriz, cuando se utiliza un lenguaje de Inteligencia Artificial, por ejemplo Prolog, se tiene que generar las reglas de producción a partir de la matriz.

Lo que nosotros hicimos en nuestro sistema experto, fue organizar el conocimiento en forma de matriz y dado que, nuestra herramienta de desarrollo (Lenguaje C++) lo permitió, nuestra base de conocimiento es en sí la estructura de matriz. Nuestra base de conocimiento es tratada como un conjunto de reglas SI-ENTONCES, donde la primera columna representa la solución y columnas posteriores indican condiciones para que dicha solución sea disparada.

Nuestra base de conocimiento está dividida en dos partes: primero, la definición de cada tipo de personalidad y segundo la definición de cada una de las 449 carreras. A continuación se describen las matrices empleadas, sus archivos y la forma en que son leídas por otros procedimientos del motor de inferencia.

Tipos de personalidad

La matriz que describe los tipos de personalidad está contenida en el archivo *base1.dat* , el cual se muestra a continuación:

PERSONALIDAD	HAB-INT	VALORES
REALISTA	8 9	9 A H
INVESTIGATIVO	6 7	6 E G I K
ARTISTICO	2 3 4	2 6 8 C
SOCIAL	0	0 3 5 J
EMPRENDEDOR	1	1 4 A D
CONVENCIONAL	5 7	7 A B F

La matriz describe a cada personalidad de acuerdo a un perfil habilidad-interés y a ciertos valores. Las características de cada tipo de personalidad están de acuerdo a la definición de las mismas dada en el apéndice B. El primer renglón del archivo muestra el nombre de la columna. La primera columna contiene el nombre de la personalidad (primeros 13 caracteres del archivo). La segunda columna (caracteres 17 al 21) contiene una clave que describe el perfil habilidad-interés requerido para dicho tipo de personalidad, las claves corresponden a las áreas de interés-habilidad como sigue:

0 - Asistencial
1 - Ejecutivo - Persuasivo
2 - Verbal
3 - Artístico - Plástico
4 - Musical
5 - Organización
6 - Científico
7 - Cálculo
8 - Mecánico - Constructivo
9 - Actividad al aire libre

La tercera columna (caracteres 32 al 40) representa aquellos valores que el usuario indicó como importantes para él en la tabla de valores (apéndice B). Las claves están organizadas como sigue:

0 - Ayudar a otros
1 - Competencia
2 - Creatividad artística
3 - Trabajar en grupo
4 - Aventura
5 - Amistad
6 - Trabajo individual
7 - Seguridad
8 - Estética
9 - Retos físicos
A - Riqueza
B - Estabilidad
C - Cambio y variedad
D - Tomar decisiones
E - Conocimiento
F - Precisión en el trabajo
G - Status intelectual
H - Trabajo práctico
I - Libertad de tiempo
J - Afiliación
K - Creatividad científica

Datos sobre las carreras

La matriz que describe a cada carrera está contenida en el archivo *carreras.dat*. Cada renglón describe a una carrera. A continuación se muestran los primeros tres renglones de la matriz de carreras:

PROFESION, RAMA Y CAMPO, AREA

ABOGADO ES 1 47 1 1

Los dos primeros renglones del archivo solo son para referencia de lo que significa cada columna. A partir del tercer renglón, la información está organizada de la siguiente forma:

a) Primeramente se describe el nombre de la carrera en un espacio de 42 caracteres.

b) Enseguida, se describe la personalidad afín a la carrera (caracteres 43 al 45). Esto se hace empleando la primera letra del tipo de personalidad de acuerdo al apéndice B. De esta forma, para nuestro ejemplo, la carrera ABOGADO requiere las personalidades Emprendedor y Social.

c) Enseguida (caracter 47 del archivo) aparece un número. Este número indica cuántas son las claves de rama y campo que contiene esa carrera. Es decir, si encontramos un número 5, los siguientes 5 números que aparezcan (cada número se separa de otro con un espacio en blanco) corresponden a 5 claves de rama y campo ocupacional de la carrera. Esto se hizo debido a que cada carrera tiene una cantidad diferente de claves de rama y campo ocupacional. Para nuestro ejemplo, aparece el número 1 por lo que solo se lee una clave de rama y campo ocupacional de la carrera, la cual es 47. La interpretación de las claves de rama y campo es en base a las definidas en el apéndice B (Catálogo de ocupaciones) de acuerdo con las siguientes equivalencias:

Catálogo de ocupaciones	Matriz de carreras	Catálogo de ocupaciones	Matriz de carreras
0.1	0	6.1	30
0.2	1	6.2	31
0.3	2	6.3	32
0.4	3	6.4	33
1.1	4	6.5	34
1.2	5	6.6	35
1.3	6	6.7	36
2.0	7	6.8	37
2.1	8	7.1	38
2.2	9	7.2	39
2.3	10	7.3	40
2.4	11	7.4	41
2.5	12	7.5	42
2.6	13	7.6	43
2.7	14	8.1	44
2.8	15	8.2	45
2.9	16	8.3	46
3.0	17	8.4	47
3.1	18	8.5	48
3.2	19	8.6	49
3.3	20	8.7	50
3.4	21	8.8	51
3.5	22	8.9	52
3.6	23	9.1	53
3.7	24	9.2	54
3.8	25	G.1	55
3.9	26	G.2	56
4.1	27	G.3	57
4.2	28		
5.0	29		

De acuerdo con lo anterior, viendo nuestro ejemplo, la carrera ABOGADO tiene el campo y rama ocupacional 47, que equivale a 8.4 en el catálogo y que se define, en el mismo catálogo, como "servicios profesionales y técnicos; de instalación de maquinaria y equipo, de protección y custodia" .

d) Después de haber leído las claves de rama y campo indicadas, aparece un número que índica la cantidad de áreas de estudio profesional para la carrera. Para nuestro ejemplo, el número indicado es el 1, por lo que a continuación se lee una clave de área de estudio profesional: 1. Dicha clave de área de estudio profesional corresponde a el área de ABOGADO según las *áreas de estudio profesional* del apéndice B.

Esta es la forma en la que se organiza la base de conocimiento de nuestro sistema experto. Podemos decir que la conforman los archivos *BASE1.DAT y CARRERAS.DAT*. En la siguiente etapa se describe como son empleados por el motor de inferencia.

III.8

Desarrollo del Software

Llegamos a la parte que consideramos más complicada de nuestra explicación de desarrollo de nuestro sistema experto: el desarrollo del software. De alguna forma ya hemos empezado a describir la codificación de nuestro programa al describir nuestra base de conocimiento en la etapa anterior. En esta etapa, entraremos más a fondo en el funcionamiento del sistema.

A la totalidad de nuestro sistema, le hemos llamado ***Doctor Vocacional***, el cual incluye al sistema experto y a la guía de carreras. La figura 3.1muestra el diagrama a bloques de nuestro sistema.

Empezaremos por describir el sistema experto para posteriormente tratar la guía de carreras.

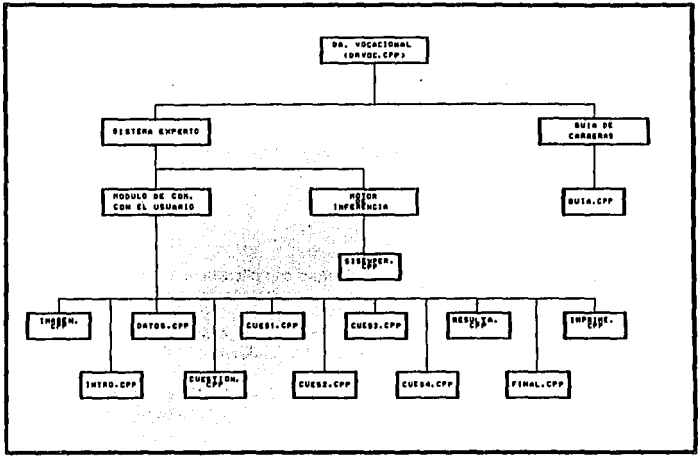

Fig. 3.1. Diagrama a bloques del Dr. Vocacional

Como se mencionó en el capítulo I, un sistema experto consta de las siguientes partes:

a) Base de conocimiento
b) Motor de inferencia
c) Módulo de comunicación con el usuario
d) Módulo de comunicación con el experto

En nuestro sistema experto fueron desarrolladas cada una de estas partes, excepto el módulo de comunicación con el experto, el cual puede ser cualquier editor de textos y el compilador Turbo C++ de Borland.

La base de conocimiento fue ya descrita, por lo que nos falta tratar el motor de inferencia y el módulo de comunicación con el usuario. Primeramente mostraremos la totalidad de los archivos que contienen los programas de nuestro sistema, incluyendo aquellos que son parte de la guía de carreras. Se muestra el

nombre del archivo, la parte del sistema a la que pertenece y los archivos de datos que emplea.

ARCHIVO	PARTE DE	ARCHIVOS QUE EMPLEA
DRVOC.CPP	MODULO DE COMUNICACION CON EL USUARIO	
IMAGEN.CPP	MODULO DE COMUNICACION CON EL USUARIO	INICIO.PCX
INTRO.CPP	MODULO DE COMUNICACION CON EL USUARIO	INTRO1.TXT, INTRO2.TXT, INTRO3.TXT, INTRO4.TXT
DATOS.CPP	MODULO DE COMUNICACION CON EL USUARIO	DATPER.TXT, DATPER.DAT
CUESTION.CPP	MODULO DE COMUNICACION CON EL USUARIO	INT3.TXT, INT31.TXT, AYUINT.TXT, INTERES.TXT, MEMORIA.DAT
CUES1.CPP	MODULO DE COMUNICACION CON EL USUARIO	INT3H.TXT, INT31H.TXT, AYUHABI.TXT, HABIL.TXT, MEMORIA.DAT
CUES2.CPP	MODULO DE COMUNICACION CON EL USUARIO	INT3V.TXT, INT31V.TXT, VALOR.TXT, MEMORIA.DAT
CUES3.CPP	MODULO DE COMUNICACION CON EL USUARIO	INT3OC.TXT, INT3OC1.TXT, INT3OC2.TXT, INT3OC3.TXT, INT3OC4.TXT, INT3OC5.TXT, INT3OC6.TXT, OCUPA.TXT, MEMORIA.DAT
CUES4.CPP	MODULO DE COMUNICACION CON EL USUARIO	INT3AE.TXT, INT3AE2.TXT, AREA.TXT, MEMORIA.DAT

ARCHIVO	PARTE DE	ARCHIVOS QUE EMPLEA
RESULTA.CPP	MODULO DE COMUNICACION CON EL USUARIO	RES1.TXT, RES2.TXT, AYUINT.TXT, AYUPER.TXT, GRAFIPER.TXT, ORDEN.TXT
FINAL.CPP	MODULO DE COMUNICACION CON EL USUARIO	FINAL1.TXT, FINAL2.TXT MEMORIA1.DAT
IMPRIME.CPP	MODULO DE COMUNICACION CON EL USUARIO	DATPER.DAT, MEMORIA.DAT
SISEXPER.CPP	MOTOR DE INFERENCIA	MEMORIA.DAT, BASE1.DAT, CARRERAS.DAT, MEMORIA1.DAT
MARCO.CPP	MODULO DE COMUNICACION CON EL USUARIO	
GUIA.CPP	GUIA DE CARRERAS	GUIA2.TXT, GUIA3.TXT, GUIA4.TXT, GUIA5.TXT, GUIA6.TXT, GUIA7.TXT, GUIA8.TXT, GUIA9.TXT, DIRECT.TXT, GCARR2.DAT, GCARR4.DAT, GCARR6.DAT, GCARR8.DAT, GCARR10.DAT, GCARR12.DAT, GCARR14.DAT, GCARR16.DAT, GCARR18.DAT

***Tabla 3.2** Archivos que conforman el sistema.*

Para un mejor entendimiento, a continuación trataremos por separado el módulo de comunicación con el usuario y el motor de inferencia. Se describen los programas que componen cada parte del sistema experto y en el apéndice A se muestran los listados completos de los mismos.

MODULO DE COMUNICACION CON EL USUARIO

Como ya se vio en el capítulo I, el módulo de comunicación con el usuario es aquel que permite la interacción usuario-computadora; se encarga de recibir los datos y de mostrar los resultados. Existe también una parte especial de este módulo que es el componente explicativo. Veamos nuestro módulo de comunicación por partes. Primeramente trataremos un conjunto de funciones que son empleadas por todos los demás programas del módulo y que están contenidas en el archivo *marco.cpp*.

ARCHIVO MARCO.CPP

Este archivo contiene las funciones necesarias para desplegar nuestras pantallas de presentación. En estas funciones se maneja la filosofía de la *programación orientada a objetos* [2]. Se definen cuatro objetos:

a) Botones
b) Ratón
c) Pantallas
d) Pantallas emergentes

Botones

La clase *botton* se usa para describir los botones que aparecen en pantalla. La clase está definida de la siguiente manera:

```
class botton  {
        int x1, y1, x2, y2;
        int fondo;
        int color;
        int sombra;
        int panta;
        char texto;
public:
        botton(int ax1, int ay1, int ax2, int ay2, int afondo, int acolor, int asombra, int
                apantalla, char atexto[8]);
        void push( int color, int tex);
        void pon( );
        void ilumina( int color, int tex);
};
```

[2] Para una mejordocumentación sobre Programación Orientada a Objetos, consúlte el libro *Turbo C/C++, Manual de Referencia*, Herbert Schildt. McGraw Hill.

Los datos que definen a *botton* son los siguientes:

int x1,y1: son las coordenadas en pantalla de la esquina superior izquierda del botón.
int x2,y2 : son las coordenadas en pantalla de la esquina inferior derecha del botón.
int fondo: es el color que tendrá el fondo del botón de acuerdo a los colores definidos por Turbo C++.
int color: es el color que tendrá el texto que contenga el botón.
int sombra: es el color de la sombra que tendrá el botón.
int panta: es el color de la pantalla en donde estará contenido el botón.
char texto[9]: es el texto que tendrá el botón.

Por otra parte, las siguientes son las funciones que manipulan los datos de los botones:

botton(int ax1, int ay1, int ax2, int ay2, int afondo, int acolor, int asombra, int apantalla, char atexto[8]);

Esta función es el constructor de la clase. Lo único que hace es inicializar los valores del botón con aquellos que recibe como parámetros.[1]

void push(int color, int tex);

La función *push*, lo que hace es simular en pantalla que el botón es oprimido. Utiliza una escritura directa en memoria de vídeo. Lo que hace es tomar el botón sin la sombra, cambiar su color por el que indica la variable *color* y cambiar el color del texto por el que indica la variable *tex*. Posteriormente guarda esta porción del botón en un buffer, limpia el área donde estaba el botón y vuelve a poner el contenido del buffer pero recorrido en dos posiciones en x.

void pon( );

Lo que hace esta función es poner el botón en pantalla. Utiliza el color de fondo y sombra definidos por el constructor así como el texto. Escribe directamente en memoria de vídeo.

void ilumina(int color, int tex);

[1] Para ver los listados de las funciones completos, remítase al Apéndice A.

La función *ilumina* cambia el color del botón y el color con el que aparece su texto de acuerdo con los parámetros *color* y *text*. También escribe directamente en memoria de vídeo.[4]

Ratón

Esta clase es empleada para utilizar el ratón. Se define como sigue:

```
class mouse {
public:
        int boton;
        int x;
        int y;
        void inicia();
        void activa();
        void desactiva();
};
```

El significado de los datos que definen a la clase *mouse* es el siguiente:

int boton: indica si algún botón del ratón fue presionado.
int x: indica la posición en x del cursor del ratón.
int y: indica la posición en y del cursor del ratón.

Veamos ahora las funciones que manipulan los datos.

void inicia();
Lo que hace esta función es habilitar el ratón. Utiliza la interrupción 33 poniendo en el registro AX el valor 001h.

void activa();
Una vez habilitado el ratón, la función *activa*, se encarga de leer los parámetros del ratón. La función activa la interrupción 33 con AX en 003h. La variable *x* almacena la columna en la que se encuentra el cursor y la variable *y* el valor del renglón. Para tener valores de 1 a 80 en *x* y de 1 a 25 en *y*, se dividen los valores regresados por la interrupción entre 8 y se les suma 1:

```
x=reg.r_cx / 8 + 1;
y=reg.r_dx / 8 + 1;
```

[4] Para mayor información sobre escritura en memoria de vídeo, consúlte el libro "*C para usuarios expertos*", Herbert Schildt, McGraw Hill.

La variable *boton* toma el valor de 1 cuando se presiona el botón de la izquierda, 2 cuando se presiona el botón derecho y 4 cuando se presiona el botón de en medio (ratón de tres botones).

void desactiva();
Deshabilita el ratón. Emplea la interrupción 33 con un valor de 002h en Ax.

Pantallas

Los objetos pantalla son empleados para la presentación de datos. Generalmente contienen botones para su manejo. La clase *pantalla* está definida como sigue:

```
class pantalla  {
        int marco;
        int fondo;
        int sombra, sc;
        char archivo [13];
        int px1;
        int py1, px2, py2;
public:
        pantalla( int x1a, int y1a, int x2a, int y2a, int marcoa, int fondoa, int sombraa,
                int sca, char archivo[13]);
        void pon_marco():
        void pon_grueso();
        void pon_delgado();
        void pon_fondo(int color, int ffondo);
        void pon_texto(int x, int y, int inicio, int fin, int color);
        void pon_texto1(int x, int y, int inicio, int fin, int color);
        void escribe(int x, int y, int color, int fondo, char cad[]);
        void pon_paleta(int y);
        void pon_modo(int modo);
        void lver( int x1, int y1, int y2, int color, int grueso);
        void lhor(int x1, int y1, int x2, int color, int grueso);
        void pon:cursor( int cx1, int cy1, int cx2, int cy2, int ccolor, int cfondo);
        void quita_cursor(int cx1, int cy1, int cx2, int cy2);
        void borra(int x1, int y1, int x2, int y2);
};
```

Los datos que utiliza este objeto son descritos a continuación.

int marco: indica el color que tendrá el borde (marco) de la pantalla.
int fondo: indica el color que tendrá el fondo de la pantalla.
int sombra: indica el color que tendrá la sombra.
int sc: al poner la sombra, los caracteres que queden bajo ella, tendrán color *sc.*
char archivo[13]: indica el nombre de un archivo que contiene texto que usará la pantalla.
int px1, py1, px2, py2: indican las coordenadas del extremo superior izquierdo. (*px1,py1*) y del extremo inferior derecho (*px2,py2*) de la pantalla.

Las funciones tienen los siguientes propósitos:

pantalla(int x1a, int y1a, int x2a, int y2a, int marcoa, int fondoa, int sombraa, int sca, char archivo[13]);
Es el constructor del objeto. Su único objetivo es asignar los valores a las variables del objeto de acuerdo con los parámetros pasados a la función.

void pon_marco():
Esta función dibuja la pantalla con su marco de acuerdo a los valores de la misma. El marco que pone a la pantalla es de dos líneas delgadas. Utiliza escritura directa en memoria de vídeo.

void pon_grueso();
Realiza lo mismo que la función anterior, solo que dibuja un marco grueso.

void pon_delgado();
Realiza lo mismo que la función anterior, solo que con un marco delgado.

void pon_fondo(int color, int ffondo);
Esta función pone un fondo detrás de la pantalla, el cual tendrá color y fondo de acuerdo a los valores pasados a la función. También escribe directamente en memoria de vídeo.

void pon_texto(int x, int y, int inicio, int fin, int color);
Despliega desde el renglón *inicio* hasta el renglón *fin* de el archivo asociado a la pantalla, en la misma a partir de la posición *x,y* con el color indicado por la variable *color.* Escribe el contenido del archivo en la pantalla desde el renglón *y* y hasta el renglón *20* de la misma.

void pon_texto1(int x, int y, int inicio, int fin, int color);
Realiza el mismo proceso que la función anterior, solo que a diferencia de ésta, *pon_texto1* permite el desplegado de partes intermedias del archivo.

void escribe(int x, int y, int color, int fondo, char cad[]);
Escribe en la pantalla en la posición *x,y* la cadena de caracteres *cad,* con el color y fondo indicados por las respectivas variables.

void pon_paleta(int y);
Utiliza la interrupción 10 para cambiar la paleta de colores.

void pon_modo(int modo);
Utiliza la interrupción 10 para poner el modo de vídeo.

void lver(int x1, int y1, int y2, int color, int grueso);
Dibuja en la pantalla una línea vertical desde el punto *x1,y1* hasta *x1,y2* con el color indicado . Cuando la variable *grueso* vale 0, la línea dibujada es delgada, de otro modo la línea será gruesa.

void lhor(int x1, int y1, int x2, int color, int grueso);
Dibuja en la pantalla una línea horizontal desde el punto *x1,y1* hasta el punto *x2,y1.* Utiliza el color especificado y si la variable *grueso* vale 0 dibuja una línea delgada, de lo contrario será una línea gruesa.

void pon:cursor(int cx1, int cy1, int cx2, int cy2, int ccolor, int cfondo);
Cambia el color y el fondo de los caracteres escritos en la pantalla desde el punto *cx1, cy1* hasta el punto *cx2, cy2* de acuerdo con las variables *ccolor* y cfondo, simulando un cursor.

void quita_cursor(int cx1, int cy1, int cx2, int cy2);
Quita el cursor puesto por la función anterior.

void borra(int x1, int y1, int x2, int y2);
Borra desde el punto *x1,y1* hasta *x2,y2* de la pantalla.

Pantallas emergentes.

Las pantallas emergentes son empleadas para el desplegado de ayuda. Permiten el desplazamiento del texto que contienen por medio del ratón y las teclas de cursor y desaparecen con el mismo. La clase se define como sigue:

```
class pantemerge  {
        int marco;
        int fondo;
        int sombra, sc;
        char archivo[13];
        int px1;
        int py1, px2, py2;
public:
        pantemerge(int x1a, int y1a, int x2a, int y2a, int marcoa, int fondoa, int
                sombraa, int sca, char archivoa[13]);
        void escribe(int x, int y, int color, int fondo, char cad[]);
        void pon_marco();
        void pon_texto(int inicio, int color);
        int control(int color);
}
```

A continuación se describen las variables que definen las clases.

int marco: indica el color del marco de la pantalla.
int fondo: indica el color del fondo de la pantalla.
int sombra: indica el color de la sombra.
int sc: indica el color con que quedarán los caracteres debajo de la sombra.
char archivo[13]: indica el nombre del archivo que contendrá la pantalla.
int px1, int py1, px2, py2: indica las coordenadas del extremo superior izquierdo y extremo inferior derecho de la pantalla.

Las funciones son las siguientes:

pantemerge(int x1a, int y1a, int x2a, int y2a, int marcoa, int fondoa, int sombraa, int sca, char archivoa[13]);

Es el constructor de la clase. Inicializa los valores de la pantalla emergente con los valores que recibe la función.

void escribe(int x, int y, int color, int fondo, char cad[]);

Escribe en la posición *x,y* de la pantalla emergente, la cadena *cad* con el color y el fondo indicado por las respectivas variables.

void pon_marco();

Dibuja la pantalla emergente de acuerdo a los valores que la definen. Al hacerlo, va dibujando desde el centro hacia las orillas, de modo que da la apariencia de que la pantalla emerge.

void pon_texto(int inicio, int color);
Despliega en pantalla 13 renglones a partir de *inicio,* del archivo asociado a la pantalla emergente. Utiliza el color indicado.

int control(int color);
Una vez desplegada la pantalla emergente, establece el control permitiendo el desplazamiento del texto con el mouse y las teclas del cursor y desapareciendo la pantalla con el uso del ratón.

Además de las funciones contenidas por los objetos ya explicados, existen las siguientes funciones en el archivo MARCO.CPP:

*void salva_pant(int x1, inty1, int x2, int y2, unsigned char *buffer);*
Esta función guarda una parte de la pantalla, indicado por *x1,y1,x2,y2;* en el puntero *buffer.* Lo que hace es primero reservar memoria para el puntero y después leer el contenido en memoria de vídeo para guardarlo en el mismo.

*void restaura_pant(int x1, inty1, int x2, int y2, unsigned char *buffer);*
Restaura la porción de función guardada en el puntero *buffer* en la posición determinada por *x1,y1, x2,y2.*

void borra(int x1, int y1, int x2, int y2);
Borra una sección de la pantalla indicada por *x1,y1, x2,y2.* Realmente lo que hace es ir a memoria de vídeo y cambiar los caracteres existentes por espacios en blanco, dejando intacto el byte de atributo.

char leetecla();
Lee una tecla y regresa:
'A' en caso de flecha hacia arriba.
'B' en caso de flecha hacia abajo.
'C' en caso de flecha a la derecha.
'D' en caso de flecha a la izquierda.
'E' en caso de enter.
'F'en caso de escape.
'G' en caso de F1.
'H' en caso de barra espaciadora.
'Q'en caso de página hacia abajo.
'I' en caso de página hacia arriba.

void cambia(int x1, int y1, int y2, int color, int fondo);
Cambia el color y fondo de una sección de pantalla indicada por *x1,y1, x2,y2.* Lo que hace es ubicar el caracter en memoria de vídeo y cambia su byte de atributos.

int leenum04(int x, int y, int colorm int fondo);
Lee del teclado números de 0 al 4 no aceptando otros caracteres. Una vez leído el número, lo escribe en pantalla en la posición *x,y* y con el color y fondo indicados por los parámetros correspondientes.

char que_hay(int x, int y);
Devuelve el caracter que hay en pantalla en la posición *x,y.* Lo que hace en realidad, es calcular la dirección en memoria de vídeo de acuerdo a las coordenadas, y leer el caracter que ahí se encuentra.

int error(int num, char archi[13]);
Esta función se encarga de desplegar mensajes de error. La variable *num,* indica que mensaje se desplegará de acuerdo al siguiente código:

1: ERROR AL ABRIR ARCHIVO
2: ERROR: NO HAY PAPEL EN IMPRESORA
3: ERROR EN IMPRESORA
4: ERROR: IMPRESORA FUERA DE LINEA

La variable *archi,* indica el nombre de archivo a desplegar cuando se presenta el error 1.

Las funciones contenidas en *maco.cpp* emplean las librerías *iostream.h, dos.h, conio.h, stdio.h, string.h, stdlib.h, ctype.h y bios.h* de Turbo C++.

Ahora que ya hemos revisado el archivo *marco.cpp,* el cual es la base para el módulo de comunicación con el usuario, veamos los demás programas que forman a éste. Podemos dividir el proceso en las siguientes partes:

a) Presentación de la portada de inicio.
b) Presentación del menú inicial.
c) Aplicación de los cuestionarios.
d) Presentación de los resultados.

Presentación de la portada de inicio

Como todo programa, nuestro sistema tiene una portada de inicio. En dicha portada presentamos una estudiante y el nombre de nuestro sistema: *Doctor Vocacional.*

La portada desplegada por el programa **IMAGEN.CPP**. Las funciones contenidas por dicho programa, permiten desplegar en pantalla una imagen guardada en formato PCX. Dado que el objetivo de esta tesis no es meterse con cuestiones de manipulación de imágenes, únicamente mostraremos el propósito de cada función. Las funciones incluidas son las siguientes:

int mainimagen();

Esta es la función que lleva el control sobre las demás. Lo que hace es ejecutar la función *pon_imagen*, indicándole que despliegue la imagen contenida en el archivo **INICIO.PCX**.

*void pon_imagen(char *archi[]);*

Esta es la función que después de *mainimagen*, invoca a todas las demás. Lleva un buen manejo de errores indicando cuando la imagen no puede ser leída, en otro caso, ejecuta la función *UnpackPcxFile* para desplegar la imagen *archi*.

*ReadPcxLine(char far *p, FILE *fp);*

Lee una línea del archivo apuntado por *fp*, y utilizando el puntero *p* despliega el contenido en pantalla.

void init();

Activa el modo gráfico empleando la interrupción 10.

void deinit();

Desactiva el modo gráfico empleando la interrupción 10.

*void UnpackPcxFile(FILE *fp);*

Utiliza la función *ReadPcxLine* para desplegar toda la imagen en pantalla. También ejecuta las funciones *sound, delay y nosound* para hacer escuchar un sonido.

*void setvgapalette(char *p);*

Inicializa la paleta de vídeo en 256 colores por medio de la interrupción 10.

*void errorimagen(char *s);*

Se encarga de invocar la función *error* de *marco.cpp* para desplegar mensajes de error.

Las funciones contenidas en IMAGEN.CPP utilizan las librerías de Turbo C++ *alloc.h, graphics.h, mem.h, conio,h* además de la utilería desarrollada *marco.h*. El archivo *marco.h* contiene las declaraciones de las funciones contenidas en *marco.cpp*.

***Fig. 3.3** . Portada inicial del Sistema Experto*

Presentación del menú inicial

La figura 3.4 muestra el menú de inicio de nuestro sistema. Tiene tres opciones:

a) Consulta del sistema experto.
b) Consulta de la guía de carreras.
c) Salir del sistema.

Cada opción puede elegirse por medio del ratón o por medio del teclado.

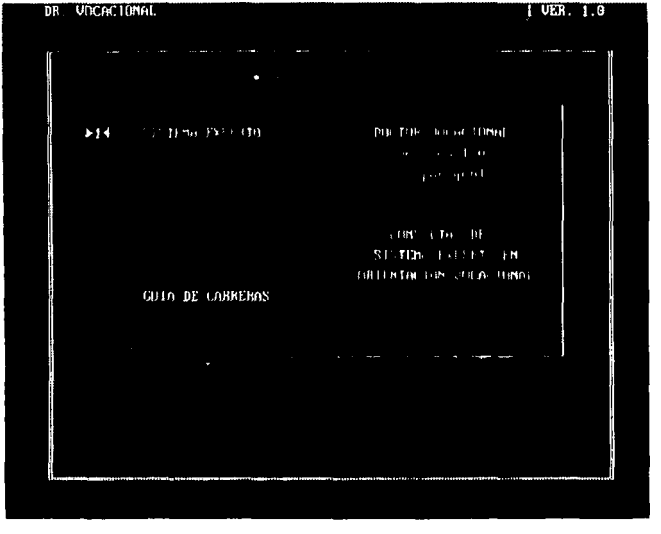

Fig. 3.4. Menú de inicio

El menú de inicio es desplegado por las funciones contenidas en el archivo DRVOC.CPP. Dichas funciones son las siguientes:

int main();

Es la función principal, sobre la cual recae todo el control del sistema. Es la única función *main* definida en los programas. Lo que hace es invocar a *mainimagen()*, la cual despliega la pantalla de presentación y enseguida transfiere el control a *menufinal()*. Por último regresa el vídeo en modo texto con fondo negro y color gris, limpia la pantalla y aparece el cursor. Le recordamos que los listados completos de las funciones se encuentran en el apéndice A.

int menufinal();

Esta función está encargada de desplegar el menú de inicio. Podemos dividir su proceso en dos partes:

1) Desplegar la pantalla de presentación.
2) Leer la opción elegida y ejecutar una acción.

La primera parte es realizada mediante la declaración de los objetos pantalla y botones. Se defina la pantalla del menú por medio de la siguiente instrucción:

pantalla a1(5,2,75,23,15,5,0,1," ");

Como se observa, la pantalla abarca desde el punto 5,2 hasta 75,23. El color de su marco es blanco (15), su fondo es violeta (5), la sombra es negra (0) y resalta los caracteres en azul (1) además no se asocia ningún archivo a ella.

Se definen también los siguientes botones:

```
botton b1(50,21,65, 22,1,3,0,5,"TERMINAR");
botton e1(9,6,14,7,1,3,0,5,"1");
botton e2(9,14,14,15,1,3,0,5,"2");
```

Estos botones aparecerán con los textos: TERMINAR, 1 y 2. Sus posiciones en la pantalla son de 50,21 a 65,22; 9,6 a 14,7 y 9,14 a 14,15. Los tres aparecen en color azul (1), sus textos en color azul cielo (3), su sombra en negro (0) y la pantalla sobre la cual están tiene color violeta (5).

Enseguida se comienza a construir la pantalla de presentación:

```
a1.pon_fondo(1,3);
a1.pon_marco();
b1.pon();
e1.pon();
e2.pon();
e1.ilumina();
```

Con estas instrucciones se pone el fondo de la pantalla, se dibuja la misma, se despliegan los botones y se ilumina el botón *e1* (opción 1, Sistema Experto). Enseguida se ponen letreros en pantalla con la función *a1.escribe y* se dibujan algunas líneas.

El proceso de lectura de una opción se hace con la estructura *do while (continua=='1')*. La variable *continua* se inicializa con valor 1. Al entrar en la estructura *do*, se pregunta si se ha oprimido alguna tecla, si es así, se checa si la

tecla es enter o movimiento de cursor. En caso de ser movimiento de cursor, se ilumina el nuevo botón hacia donde se desplaza el cursor y se dibuja el botón antes iluminado . Cuando la tecla es enter se verifica el valor de la variable *xpos,* por medio de la cual se sabe cual es el botón actual. De acuerdo con esto, se ejecuta alguna de las dos funciones, *sistema_experto(),* o *mainimagen().*

Cuando se selecciona la opción salir del sistema, se ejecuta la instrucción *return(1).*

Además de checar el teclado, se ejecuta la instrucción *ratón.activa(),* la cual permite sensar los parámetros del ratón. Cuando se ha oprimido el botón izquierdo del ratón *(if (ratón.boton==1))* se verifican los valores de *x* y *y* del mismo, y en caso de coincidir con alguna de las áreas de los botones, se ejecuta la acción correspondiente.

Cuando se elige alguna opción, se da la apariencia de que el botón correspondiente es oprimido, por medio de la función *push.* También, antes de ejecutar otra función, se salva la pantalla por medio de *salva_pant,* y al regresar dicha función el control del programa, la pantalla es restaurada por medio de *restaura_pant.*

A lo largo del sistema experto, se manejan funciones del tipo de *menufinal().* Dichas funciones, lejos de manejar algoritmos complicados, lo único que hacen es construir pantallas de presentación, checar teclas y mouse y de acuerdo con los valores obtenidos de estos periféricos, realizar acciones como iluminar o restaurar botones y ejecutar funciones. Creemos que eso es fácil de entender aunque el código sea un poco extenso. En lo consecuente se explican las funciones de manera superficial y se enfatiza en el orden en que se ejecutan los procesos y en las estructuras donde se guardan los datos recopilados por el *módulo de comunicación con el usuario.*

Aplicación de los cuestionarios

La aplicación de los cuestionarios es la parte medular del módulo de comunicación con el usuario. Los cuestionarios que se aplican son seis:

a) Datos personales.
b) Cuestionario de intereses.
c) Cuestionario de habilidades.
d) Tabla de valores.
e) Catálogo de ocupaciones.

f) Areas de estudio profesional.

Sin embargo, antes de empezar los cuestionarios se despliega una introducción. A continuación se explica cómo se realizan dichas tareas.

INTRODUCCION INICIAL

Antes de comenzar con el sistema experto, se muestra al usuario una introducción. La figura 3.5 muestra la pantalla de dicha introducción.

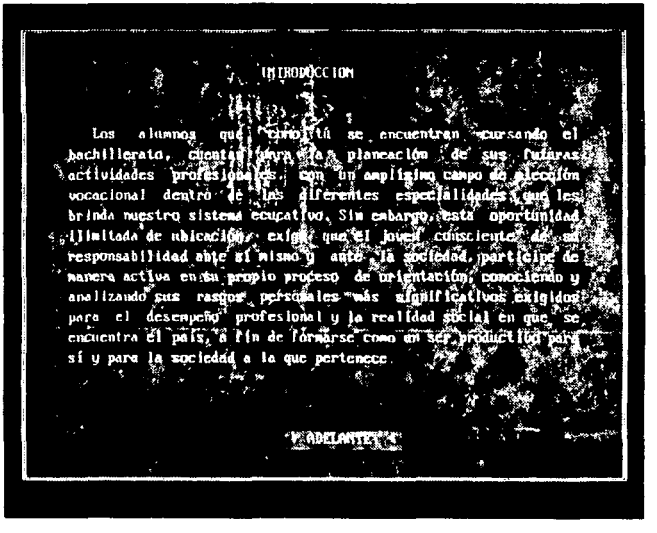

Fig. 3.5. Introducción al sistema experto.

La introducción es desplegada por el programa INTRO.CPP. Dicho programa contiene la función *introduce*, declarada de la siguiente manera:

int introduce (char archi [13]);

Esta función despliega en una pantalla como se muestra en la figura 3.4. Tiene las opciones *ADELANTE* y *Pág. Ant.* . Dichas opciones son utilizadas para continuar con la introducción o regresar a la página anterior. Esto se controla mediante estructuras *while* en la función *main_intro();* aprovechando que cuando se presiona el botón de *Pág. Ant. introduce* regresa un 0.

La introducción es leída de los siguientes archivos de texto: *INTRO1.TXT, INTRO2.TXT, INTRO3.TXT* e *INTRO4.TXT.*

CUESTIONARIO DE DATOS PERSONALES

Una vez desplegada la introducción, el módulo de comunicación con el usuario comienza su tarea de recopilar datos del usuario. Lo primero que pide son los datos personales.

El cuestionario de datos personales es mostrado en la figura 3.6. Los datos que pide son : nombre, fecha de nacimiento, edad, sexo, estado civil, escolaridad, ocupación, domicilio y teléfono. El cursor se desplaza con la tecla de tabulación, se permite la corrección de los datos y una vez que el usuario ha introducido los mismos debe presionar el botón *ACEPTAR.*

El cuestionario es aplicado por las funciones en el archivo DATOS.CPP. Dichas funciones son las siguientes:

int pon(int x, int y, char car, int pos);

Esta función únicamente pone en pantalla un caracter en color blanco (15) y con fondo negro (0). Es empleada para desplegar en pantalla los caracteres que son leídos.

int datos();

Realmente es esta función la que lleva todo el peso de la aplicación del cuestionario. Primeramente se encarga de desplegar la pantalla de presentación y leer los datos, así como llevar el control del desplazamiento del cursor. Los datos son almacenados en las variables siguientes:

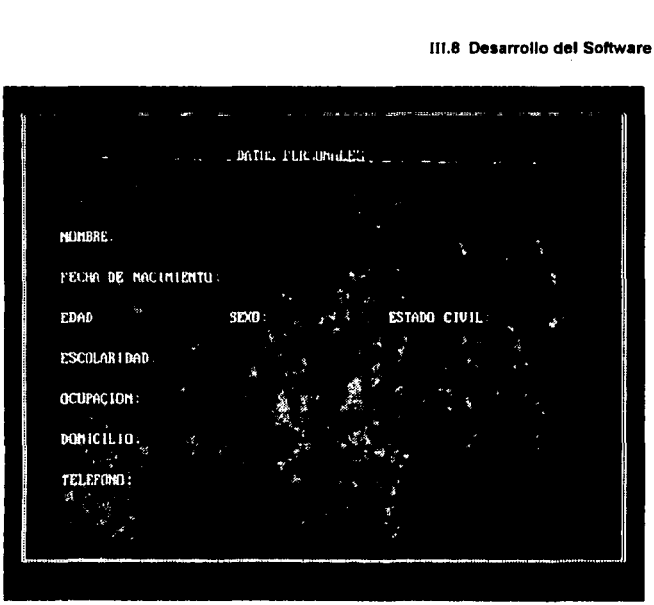

Fig. 3.6. *Datos personales.*

```
char nombre [41];
char fecha [9];
char edad [3];
char sexo [11];
char estado [11];
char escolaridad [21];
char ocupa [41];
char domicilio [48];
char telefono [11];
```

Una vez leídos los datos, estos son almacenados en el archivo DATPER.DAT mediante las siguientes instrucciones:

fprintf(fp, "%s\n%s\n%s\n%s\n%s\n", nombre, fecha, edad, sexo, estado);
fprintf(fp, "%s\n%s\n%s\n%s\n", escolaridad, ocupa, domicilio, telefono);

De acuerdo con lo anterior, los datos son guardados cada uno en un renglón, es decir, cada variable está separada de la otra por medio de un caracter \n (enter).

CUESTIONARIO DE INTERESES

El siguiente cuestionario que aplica nuestro sistema experto es el cuestionario de intereses. Antes de empezar, se le da al usuario una introducción y una explicación de cómo contestar a las preguntas. La figura 3.7 muestra la pantalla de presentación del cuestionario de intereses.

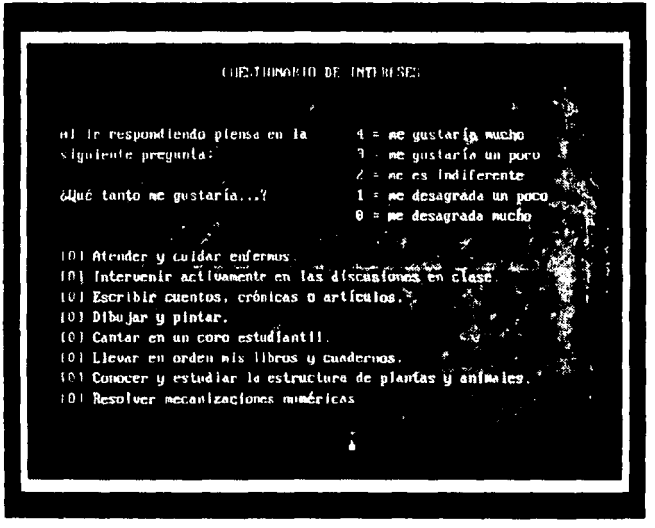

Fig. 3.7. Cuestionario de intereses.

La pantalla de presentación de este cuestionario muestra un cursor para contestar a cada pregunta, en donde sólo se aceptarán como posibles respuestas los números del 0 al 4. El cursor se desplaza con las teclas de cursor. Se dispone también con dos botones, uno para avanzar en el cuestionario y otro para retroceder. Las respuestas dadas pueden ser modificadas poniendo el cursor en el lugar correspondiente y pulsando la tecla deseada.

El cuestionario de intereses es manipulado por las funciones contenidas en el archivo CUESTION.CPP. Dichas funciones son las siguientes:

int main_intereses();

Esta es la función que controla la aplicación del cuestionario. Primeramente invoca funciones para el desplegado de una introducción, después aplica el cuestionario, muestra algunos resultados parciales y almacena los datos obtenidos.

int introducc1();

La introducción al cuestionario de intereses está contenida en dos archivos. *Introducc1()* despliega la primera parte contenida en el archivo INT3.TXT. Esta función se encarga de desplegar la pantalla de presentación para la introducción y controla el curso del cuestionario por medio de las dos teclas *ADELANTE* y *Pág. Ant.*

int introducc2();

Despliega la segunda parte de la introducción contenida en el archivo INT31.TXT. Maneja el control del cuestionario por medio de dos botones que permiten avanzar o retroceder en el mismo. Al igual que *introducc1()*, permite el manejo del ratón y del teclado.

char cambia_panta(pantalla c1, botton b1, botton b2, int interes[60], int pasada);

Las preguntas que se aplican en el cuestionario de intereses están contenidas en el archivo *INTERES.TXT*. Dado que son 60 preguntas, éstas no pueden mostrarse en una sola pantalla, por lo que es necesario cambiar las mismas. *Cambia_panta* realiza esa función, basándose en el parámetro *pasada* para saber que porción del archivo va a desplegar. Una vez que cambia el contenido de la pantalla, sede el control a la función *lee_interes*.

char lee_interes(pantalla c1, botton b1, botton b2, int interes[60], int pasada);

Esta función lee las respuestas para cada una de las preguntas del cuestionario. Los datos leídos son almacenados en el arreglo:

int interes[60];

Permite el desplazamiento del cursor con el ratón y con las teclas del cursor. Permite también el desplazamiento del cuestionario hacia atrás o hacia adelante regresando '1' cuando se oprime el botón ADELANTE y '0' cuando se oprime el botón Pág. Ant.

void cuest1 (int intereses [60]);

Esta función controla realmente la aplicación del cuestionario. Primeramente construye la pantalla de presentación del cuestionario, limpia el arreglo en donde se almacenarán los valores y controla las llamadas a *cambia_panta()*.

void evalint(int interes[60], int evinter[10]);

Una vez capturadas las respuestas a las preguntas del cuestionario, se debe hacer una suma para sacar la puntuación de cada área de interés. Recordemos que los valores de las 60 preguntas quedaron almacenados en el arreglo *intereses*. La puntuación para cada área de interés queda almacenada en el arreglo,

int evinter [10];

Debemos recordar que cada área de interés es evaluada sumando los valores de ciertas preguntas. Por ejemplo, la primer área, llamada *Asistencial,* es evaluada con la suma de las preguntas 1, 11, 21,31, 41 y 51. Debemos indicar también que el valor de la pregunta 1 está contenida en el elemento 0 del arreglo *interes*, y que el valor del área Asistencial está almacenado en el elemento 0 del arreglo *evinter*. Los elementos de *evinter* van desde 0 hasta 9 y contienen los valores de las siguientes áreas en este orden: *Asistencial, Ejecutivo-persuasivo, Verbal, Artístico-plástico, Musical, Organización, Científico, Cálculo, Mecánico-constructivo y Actividad al aire libre.*

La función *evalint* calcula la puntuación para cada área por medio de las siguientes instrucciones:

```
for(j=0; j<10; j++)
   for(i=0; y<60; y+=10)
   evinter[j]+=interes[i+j];
```

int graficaint(int evinter[10]);

Una vez que ha sido calculada la puntuación para cada área de interés, se está en posibilidad de presentar el perfil de interés. La función *graficaint*, muestra la gráfica del perfil de interés (figura 3.8).

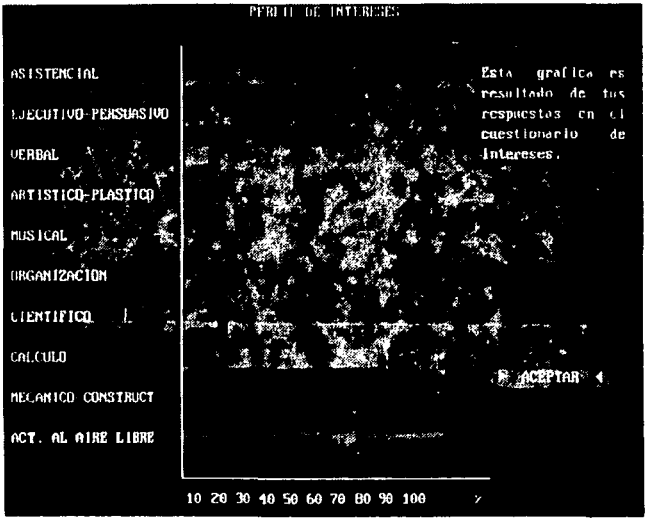

Fig. 3.8 Gráfica del perfil de intereses.

Lo que hace graficaint es tomar los valores del arreglo *evinter*, y en base a ello realizar la gráfica.

Se cuenta también con una ventana de ayuda para tener más información acerca de las áreas de interés como se muestra en la figura 3.9.

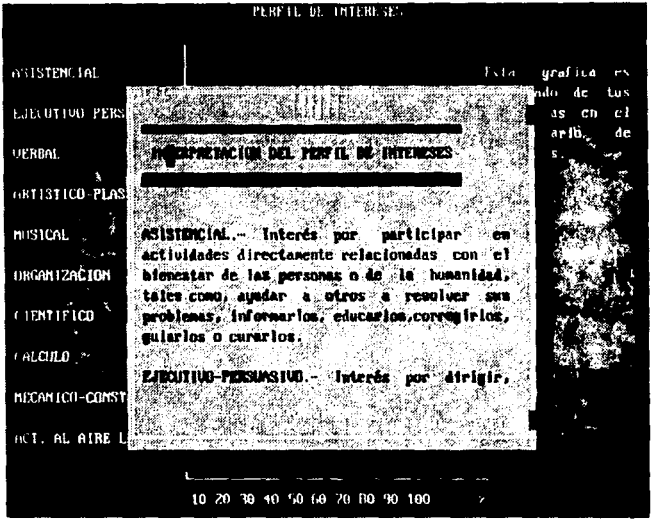

Fig. 3.9 *Ventana de ayuda en el perfil de intereses.*

int ordenaint (int evinter[10]);

Una vez obtenido el arreglo de puntuaciones *evinter*, es necesario obtener las tres áreas como mejor puntuación. Esta función la hace *ordenaint*. Primeramente se declara un arreglo de la siguiente forma:

```
typedef struct  {
        char elemento [23];
        int puntos;
}  arreglo;
arreglo intereses[10];
```

Después, se vacía el contenido del arreglo *evinter* en la variable *puntos* del arreglo *intereses* y en el parámetro *elemento se* copia el nombre de cada área. Posteriormente se ordena el arreglo *intereses* por el método de la burbuja y se muestran las tres áreas con mayor puntuación (figura 3.10).

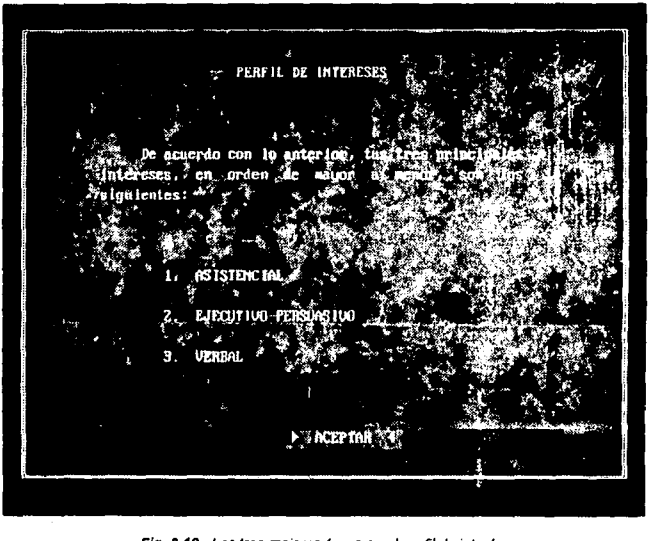

Fig. 3.10. Las tres mejores áreas en el perfil de interés.

int almacena(int evinter[10]);

Una vez realizado todo el proceso de intereses, es importante almacenar los datos obtenidos. La función *almacena*, guarda el contenido de *evinter*, el arreglo que contiene la puntuación para cada área de interés, en el archivo *MEMORIA.DAT*. Cada número del arreglo es separado de otro por un espacio en blanco y al final se inserta un caracter '\n':

```
for(i=0; y<10; y++)
   fprintf(fp, "%d ", evinter[i]);
fprintf(fp,"\n");
```

int resultaint();

Esta función es invocada cuando se ejecuta el menú final del sistema experto. En sí lo que hace es leer los datos de *evinter* del archivo *mamoria.dat* y ejecutar las funciones que muestran el perfil de intereses: *graficaint* y *ordenaint*.

En conclusión, el cuestionario de intereses, implica los siguientes archivos:

cuestion.cpp: es el código que maneja la aplicación.
int3.txt, int31.txt: contienen el texto de introducción.
interes.txt: contiene las preguntas del cuestionario.
ordenint.txt:: contiene texto para una pantalla de presentación.
ayuint.txt: contiene la ayuda para el perfil de intereses.
memoria.dat: contiene los datos obtenidos del cuestionario.

CUESTIONARIO DE HABILIDADES

Una vez explicado el proceso de aplicación del cuestionario de intereses, será más fácil explicar la aplicación de los demás cuestionarios, ya que el proceso es similar.

El cuestionario de habilidades es aplicado por las funciones contenidas en el archivo *CUES1.CPP*. El proceso es igual a el de la aplicación del cuestionario de intereses. Primeramente se presenta una introducción al cuestionario (funciones *introducc1h* e *introducc2h*). Los archivos que contienen dicha introducción son *int3h.txt* e *int31h.txt*. Se aplica el cuestionario de la misma forma que el de intereses. El archivo que contiene las preguntas es *habil.txt*. Las áreas de habilidad son evaluadas de acuerdo a lo obtenido en el cuestionario y en el mismo orden que en el cuestionario de interes, orden también guardado por el

manual, se almacenan las puntuaciones en el segundo renglón del archivo *memoria.dat*.

Se muestra al usuario su perfil de interés de la misma forma que en el cuestionario anterior y se cuenta así mismo con una ventana de ayuda.

En resumen, se utilizan los siguientes archivos:

cues1.cpp: contiene el código para la aplicación del cuestionario.
int3h.txt, int31h.txt: contienen la introducción.
habil.txt: contiene las preguntas al cuestionario.
ordenhab.txt: contiene texto para una pantalla de presentación.
ayuhabi.txt: contiene la ayuda para el perfil de habilidades.
memoria.dat: contiene los datos recabados.

TABLA DE VALORES

La siguiente tarea que realiza el módulo de comunicación con el usuario es la aplicación del cuestionario de valores o tabla de valores. Se despliega una introducción y una explicación de como contestar. El cuestionario de valores que se aplica es el que se muestra en el apéndice B y está contenido en el archivo *valor.txt*. Ahora solo hay que marcar con la barra espaciadora aquellos valores que le interesan al usuario. La pantalla de presentación del cuestionario se muestra en la figura 3.11.

Después de aplicar el cuestionario se guardan los datos en el archivo *memoria.dat*. Los valores que fueron marcados por el usuario aparecerán como un caracter ■, mientras que aquellos que no lo fueron aparecerán como un espacio en blanco.

Los archivos utilizados son los siguientes:

cues2.cpp: contiene el código que maneja el cuestionario.
int3v.txt, int3v1.txt: contiene la introducción.
valor.txt: contiene el cuestionario.
memoria.dat: contiene los datos obtenidos.

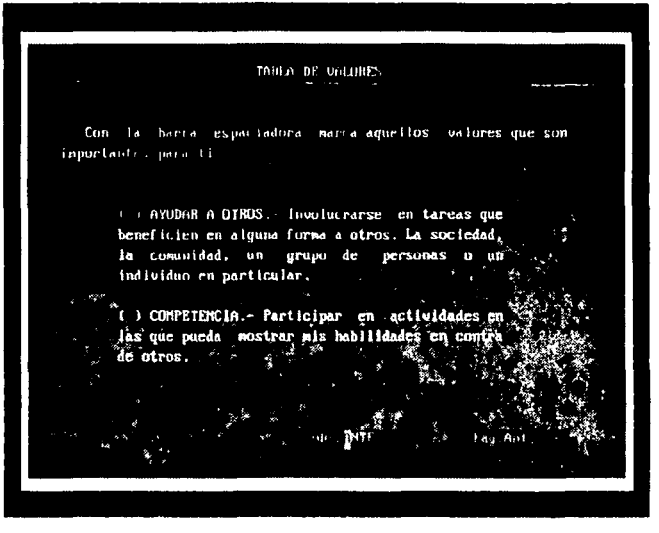

Fig. 3.11. Aplicación de la tabla de valores.

CATALOGO DE OCUPACIONES

El cuestionario correspondiente al catálogo de ocupaciones es ejecutado por el programa *CUES3.CPP*. De igual forma que los cuestionarios anteriores, se presenta antes que nada una introducción. Posteriormente se aplica el cuestionario en donde el usuario tendrá que hacer sus selecciones utilizando la barra espaciadora. Las respuestas a las 59 preguntas se almacenan en el archivo *memoria.dat*, en donde una rama seleccionada se marca con el caracter ■.

Los archivos utilizados son los siguientes:

cues3.cpp: contiene el código para la aplicación del cuestionario.
int3oc.txt, int3oc1.txt, int3oc2.txt, int3oc3.txt, int3oc4.txt, int3oc5.txt, int3oc6.txt: contienen la introducción.
ocupa.txt: contiene las ramas y campos del cuestionario.
memoria.dat: almacena los datos obtenidos.

AREAS DE ESTUDIO PROFESIONAL

La última parte del proceso mediante el cual el módulo de comunicación con el usuario recibe datos es la aplicación del cuestionario de áreas de estudio profesional. Aquí se marcan las áreas que al usuario le parezca conveniente utilizando la barra espaciadora. El cuestionario está contenido en el archivo *area.txt.* Las funciones que manejan la aplicación están en el archivo *CUES4.CPP.* Los datos obtenidos por el cuestionario se almacenan en el quinto renglón del archivo *memoria.dat.*

Los archivos empleados son los siguientes:

cues4.cpp: contiene el código que maneja el cuestionario.
int3ae.txt, int3ae2.txt: contiene la introducción al cuestionario.
area.txt: contiene las áreas de estudio que utiliza el cuestionario.
memoria.dat: almacena los datos obtenidos por el cuestionario.

En conclusión, los datos que el usuario ingresa mediante el módulo de comunicación con el usuario, son almacenados en el archivo *memoria.dat,* el primer renglón de este archivo contiene la puntuación de las diez áreas de interés, separando cada número de otro por medio de un espacio en blanco, el segundo renglón contiene la evaluación de las diez áreas de habilidad; el tercero, la respuesta a las 22 preguntas de la tabla de valores, el cuarto renglón contiene la respuesta a las 59 ramas del catálogo de ocupaciones y por último se encuentran en otro renglón la selección de las 41 áreas de estudio. Un ejemplo del archivo *memoria.dat* es el siguiente:

12 5 4 0 4 4 10 20 6 2
14 13 5 0 3 7 8 24 14 0

Presentación de los resultados.

Después de la introducción de los datos requeridos por el sistema, se realiza el proceso de inferencia para posteriormente mostrar los resultados.

El primer resultado, además del perfil de interés y el perfil de habilidad, que exhibe el módulo de comunicación con el usuario es el perfil conjunto interés-habilidad, el cual es proporcionado por el motor de inferencia. En realidad lo que se hace es mostrar las tres mejores parejas interés-habilidad, es decir, aquellas áreas en donde hay una mejor puntuación y equilibrio en los dos factores: interés y habilidad. Además de mostrar las tres mejores áreas, se le muestra al usuario una gráfica como la de la figura 3.12.

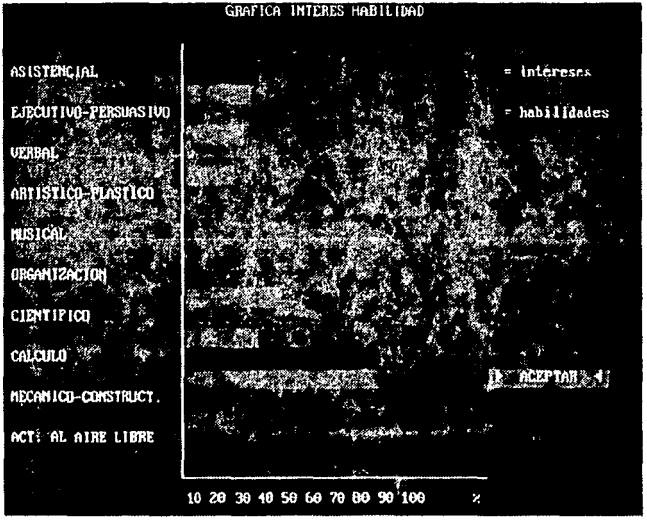

Fig. 3.12. Gráfica del perfil interés-habilidad.

Algunas de las funciones para mostrar resultados se encuentran en el archivo *RESULTA.CPP*. De este archivo, la función *grafica_inter_y_habi()* realiza la gráfica, mientras *ponmejorih()* despliega las tres mejores parejas interés-habilidad dadas por el motor de inferencia.

Otro importante resultado es el tipo de personalidad del individuo, el cual es obtenido por el motor de inferencia en base a los tres primeros cuestionarios. La función *graficaper* del archivo *resulta.cpp*, muestra el grado en que el usuario se acerca a cada tipo de personalidad con una gráfica como la mostrada en la figura 3.13.

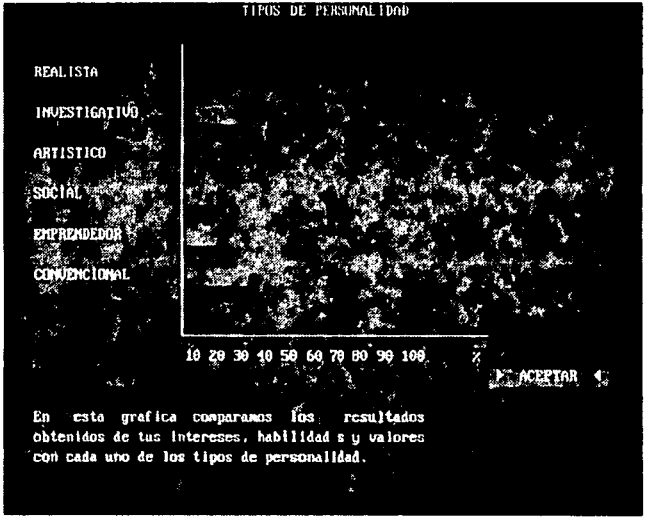

Fig. 3.13. Gráfica de los tipos de personalidad.

Por otro lado, una parte de la función *ordenaper* del archivo *SISEXPER.CPP* muestra los tres tipos de personalidad a los que más se acerca el usuario.

Los resultados finales son desplegados por las funciones contenidas en el archivo *FINAL.CPP*. En primer lugar, la función *introducc1final,* despliega una introducción que está contenida en los archivos *final1.txt* y *final2.txt*. Después, la función *presentafin* muestra las cinco carreras a las que más se acerca el perfil del usuario. La pantalla mostrada por *presentafin* se muestra en la figura 3.14.

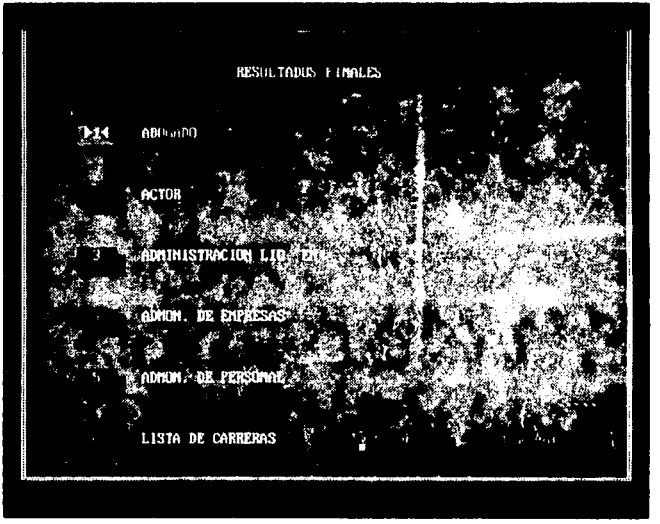

Fig. 3.14. *Resultados finales.*

Realmente, lo que hace *presentafin* es abrir el archivo *MEMORIA1.DAT,* el cual es creado por el motor de inferencia y contiene el nombre de cada carrera asociado a un coeficiente que indica que tanto se acerca el perfil del usuario a dicha carrera y un número que indica la posición de la misma en el archivo *CARRERAS.DAT*. Es entonces del archivo MEMORIA1.DAT de donde *presentafin* obtiene las cinco mejores opciones para el usuario.

Cuando en el menú de la figura 3.14, se elige una carrera, se pasa el control del programa a la función *muestra,* la cual se declara como sigue:

int muestra (int nocar, char mejorper[4]);

La variable *nocar* indica la posición de la carrera en el archivo CARRERAS.DAT, el cual contiene la descripción de cada carrera. El arreglo *mejorper* , contiene los tres tipos de personalidad a los cuales más se acerca el usuario. Esto es importante, ya que *muestra* viene a ser el **componente explicativo** de nuestro sistema experto; dadas las soluciones, *muestra* despliega el perfil requerido por la carrera y el perfil del usuario, justificando así el que se le haya prestentado dicha carrera como opción para estudiarla.

La figura 3.15 muestra la pantalla de nuestro componente explicativo.

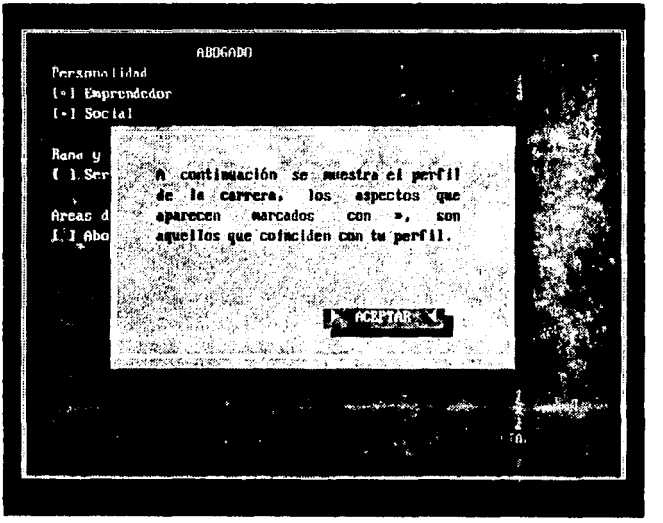

Fig. 3.15. Pantalla de explicación.

Además de mostrar lo que el sistema experto considera las cinco mejores carreras para el usuario, nuestro menú de la figura 3.13 tiene una opción para mostrar la lista de las 449 carreras contenidas en *CARRERAS.DAT*. La función *presentatodas* muestra una pantalla con la lista de las carreras y un coeficiente mostrando que tanto, el usuario, cumple con el perfil. También es posible seleccionar cada carrera para ver la justificación del valor de su coeficiente, figura 3.16.

Fig. 3.16. *Lista de carreras.*

Para terminar con nuestro módulo de comunicación con el usuario, diremos que se muestra al final del proceso del sistema experto, un menú con los principales resultados. Dicho menú lo ejecuta la función *menufinal* del archivo *SISEXPER.CPP*. El menú se muestra en la figura 3.17.

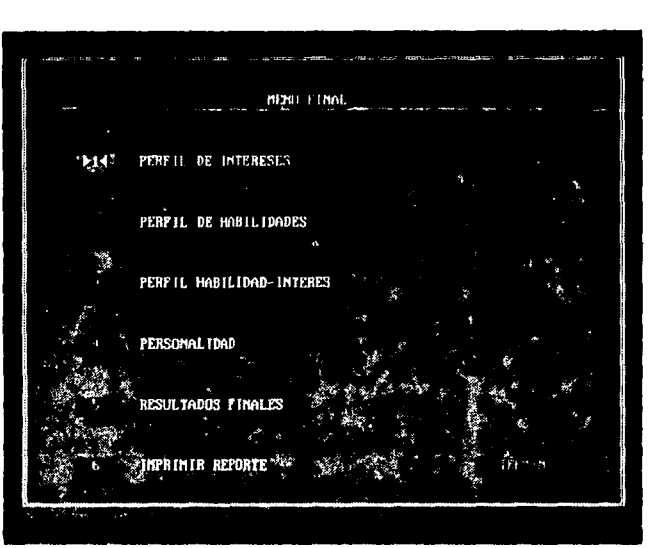

Fig. 3.17. Menú final del sistema experto.

Además, nuestro módulo de comunicación con el usuario permite obtener los resultados impresos mediante las funciones contenidas en el archivo *IMPRIME.CPP*. Se imprimen los datos personales, el perfil de interés, el perfil de habilidad, los tipos de personalidad y las diez carreras a las que más se acerca el perfil del usuario.

EL MOTOR DE INFERENCIA

Hemos llegado a la parte esencial de nuestro sistema experto: el motor de inferencia. Las funciones que lo componen están contenidas en el archivo *SISEXPER.CPP*.

Podemos dividir la labor del motor de inferencia en las siguientes tres partes:

a) Obtención del perfil interés-habilidad.
b) Obtención del tipo de personalidad.
c) Obtención de la solución final (carreras a estudiar).

De alguna forma, los procesos de ordenación del arreglo de intereses (*evinter*) y del arreglo de habilidades (*evhabi*) son parte también del sistema experto. Estos procesos fueron descritos anteriormente.

Obtención del perfil interés-habilidad.

Recordemos que al aplicar el módulo de comunicación con el usuario el cuestionario de intereses, se evalúan las áreas de interés y su valor queda almacenado en un arreglo llamado *evinter*. Lo mismo sucede con el cuestionario de habilidades, creándose el arreglo *evhabi*. Dados estos datos, la función *ordena* del archivo SISEXPER.CPP obtiene el perfil interés-habilidad.

Primeramente se declara un arreglo para almacenar las diez áreas, guardando una clave del interés o habilidad (I0, I1, ... para intereses y H0, H1, ... para habilidades) y su puntuación, la declaración del arreglo es como sigue:

```
typedef struct {
        char elemento [3];
        int puntos;
} arreglo;
arreglo intereses [10];
arreglo habilidades [10];
```

Al vaciar la puntuación de *evinter* y *evhabi* en los arreglos *intereses* y *habilidades*, respectivamente, éstos son ordenados en forma ascendente de acuerdo a su puntuación. Ahora que los arreglos ya están ordenados, se procede a elegir las tres mejores parejas interés-habilidad. De acuerdo con nuestros expertos humanos, esto debe hacerse de acuerdo a los siguientes criterios:

a) Elegir el mejor interés que es también la mejor habilidad.

b) Si el mejor interés no es también la mejor habilidad, elegir el 2o. mejor interés que es la mejor habilidad.

c) Si ninguno de los casos anteriores se presentan, elegir el 2o. mejor interés que es la 2a mejor habilidad.

d) Elegir la mejor habilidad que es el mejor interés.

e) Continuar eligiendo las tres mejores parejas interés-habilidad considerando que entre las tres opciones deben estar: el primer interés, el 2o. interés y la mejor habilidad.

Para lograr que la elección sea en base a estos criterios dictados por nuestros expertos, la función *ordena* ejecuta las siguientes líneas:

```
i=0;
if (intereses[0].elemento[1] == habilidades[0].elemento[1])
{
      mejorih[i]=intereses[0].elemento[1];        // mejor interes y mejor habilidad
      i++;
      pasa=1;
}
if ( ( intereses[1].elemento[1] == habilidades[0].elemento[1]
   || (intereses[1].elemento[1] == habilidades[1].elemento[1]))
{
      mejorih[i]=intereses[1].elemento[1];    // 2o mejor interes y mejor habilidad
      i++;                                    // o 2o interes y 2a habilidad
}
if ( i== 2)
{
      mejorih[i]=intereses[2].elemento[1];    // tercer mejor interés
      i++;
}
if ( i==1)
{
      if ( pasa ==1 )     // ya seleccionó el mejor interés
      {
            mejorih[i]=intereses[i].elemento[1];    // 2o. mejor interés
            i++;
      }
      else
      {
            mejorih[i]=intereses[0].elemento[i];    // mejor interés
            i++;
      }
      if(pasa==1)
      {
            mejorih[i]=intereses[2].elemento[1];    // 3er mejor interes
            i++;
```

```
        }
        else
        {
                mejorih[i]= habilidades [0].elemento[1];    // mejor habilidad
                y++;
        }

}
if ( i==0)
{
        mejorih[i]=intereses[0].elemento[1];    // mejor interes
        i++;
        mejorih [i ] = intereses[1].elemento[1];    // 2o mejor interés
        i++;
        mejorih [i ]=habilidades[0].elemento[1];  // mejor habilidad
        i++;
}
mejorih[3]='\0';
```

El resultado queda almacenado en el arreglo *mejorih:*

```
char mejorih[4];
```

Se almacena únicamente el número del área de interés habilidad (números del 0 al 9).

Como se puede observar, en esta parte del motor de inferencia se emplea una *búsqueda guiada por atributos*, es decir, *un encadenamiento hacia adelante.*

Obtención del tipo de personalidad

Esta labor la realizan las funciones *obten_persona* y *ordenaper* del archivo SISEXPER.CPP.

Primeramente, dadas las tres mejores parejas interés-habilidad, contenido en el arreglo *mejorih,* y las respuestas al cuestionario de valores contenido en el arreglo *valor, obten_persona* calcula un coeficiente que indica que tanto se acerca el usuario a cada tipo de personalidad. Los datos de los valores se encuentran en el archivo *memoria.dat* y la descripción de cada tipo de personalidad se encuentra en *BASE1.DAT.* El contenido de este último archivo es el siguiente:

PERSONALIDAD	HAB-INT	VALORES
REALISTA	8 9	9 A H
INVESTIGATIVO	6 7	6 E G I K
ARTISTICO	2 3 4	2 6 8 C
SOCIAL	0	0 3 5 J
EMPRENDEDOR	1	1 4 A D
CONVENCIONAL	5 7	7 A B F

Recordemos que como parte de la base de conocimiento, este archivo ya fue descrito anteriormente.

Obten_persona declara un arreglo como el siguiente:

```
typedef struct  {
        char per[14];
        int coef;
}  personalidad;
personalidad persona [6];
```

El arreglo persona guarda el nombre de la personalidad y su coeficiente. Primeramente se lee de cada línea (una línea es un tipo de personalidad) la cantidad de parámetros que la definen; es decir, cuantas parejas intereses-habilidad y cuantos valores. Por ejemplo, la personalidad *realista* está definida por cinco parámetros. Este valor es almacenado en la variable *total.* Los valores de *total* se obtienen con el siguiente código:

```
total=0;
//limpiar el arreglo
for(i=0; i<41; i++)
        linea[i]= ' ';
linea[40]='\0';
for(i=1; i<=41; i++)
{
        fscanf(fp, "%c",&c);
        if((i>13)&&(c!='\n'))
                if(c!=' ')
                        total++;
        if(c!='\n')
                linea[i-1]=c;
        else
                i=41;
}
```

Una vez que sabemos cuántos parámetros definen a cada tipo de personalidad, debemos ver con cuántos cumple el usuario. El motor de inferencia va leyendo los parámetros de la personalidad y si este se encuentra en alguno de los arreglos *mejorih* o *valor,* se incrementa la variable *estan.* Cuando la pareja interés-habilidad que define a la personalidad es la mejor del usuario (aquella que obtuvo mayor puntuación), *estan* se incrementa dos veces, con esto se le da más peso a la mejor área interés-habilidad provocando que la aproximación al tipo de personalidad sea mejor.

Finalmente, el coeficiente para cada tipo de personalidad se calcula como:

*coefi=estan*100/(total+1);*

Los coeficientes se almacenan en el arreglo *valorper* en el orden en que aparecen los tipos de personalidad en la base de conocimiento.

La función *ordenaper* toma los coeficientes de cada tipo de personalidad y los ordena, obteniendo los tres tipos de personalidad con puntuación más alta y mostrándolos al usuario. Los tres mejores tipos de personalidad quedan en el arreglo *mejor per* , declarado como sigue:

char mejorper[4];

Podemos observar que para la obtención de los tipos de personalidad, se emplea la *búsqueda guiada por atributos* o *encadenamiento hacia atrás,* ya que se inspecciona cada uno de los tipos y se asigna el coeficiente de acuerdo a los datos obtenidos por el módulo de comunicación con el usuario.

Obtención de la solución final

En este punto ya se cuenta con los elementos necesarios para poder llegar a la solución final del sistema, la cual es mostrar al usuario las cinco carreras consideradas como mejores opciones para él. Contamos con la personalidad obtenida por el motor de inferencia y con las respuestas a el catálogo de ocupaciones y áreas de estudio, proporcionadas por el módulo de comunicación con el usuario.

La función *motor1* de SISEXPER.CPP realiza un trabajo similar al hecho para obtener el tipo de personalidad: se calcula un coeficiente para cada una de las 449 carreras. Para el cálculo del coeficiente se emplean los archivos *memoria.dat,* contiene los datos del usuario, y *carreras.dat* que es parte de la base de conocimiento y que contiene la descripción para cada una de las carreras.

El cálculo del coeficiente se hace en base al siguiente porcentaje:

personalidad - 60%
rama y campo ocupacional - 20%
área de estudio profesional - 20%

Primero se verifica la personalidad que define a la carrera. El coeficiente se incrementa de acuerdo a el lugar que tenga el tipo de carrera en el arreglo *mejorper;* es decir, si es el tipo de personalidad con puntuación más alta, el coeficiente se aumenta en 40, si es el segundo lugar se aumenta en 30 y si es el tercer lugar se aumenta en 20. Se hace también una verificación, de manera que el coeficiente no sobrepase los 60 puntos:

```
if (( mejorper[j]==per[0]) || ( mejorper[j]==per[1]) || ( mejorper[j]==per[2]))
      if(j==0)
            carrera[i].coefi+=40;
      else
            if(j==1)
                  carrera[i].coefi+=30;
            else
                  if(j==2)
                        carrera[i].coefi+=20;
if(carrera[i].coefi>60)    carrera[i].coefi=40;
```

Enseguida se analizan las ramas y campos de ocupación. La variable *tot* contiene el total de ramas que definen a la carrera, mientras que *hayocupa* almacena con cuántas cumple el usuario. Entonces el coeficiente se calcula con la siguiente instrucción:

```
carrera[i].coefi+= (hayocupa*10/tot)*2;
```

Para terminar con el cálculo de coeficiente, se toma en cuenta las área de estudio. De la misma forma, *tot* almacena el número de áreas de estudio que definen a la carrera y *hayarea* el número de aquellas con las que cumple el usuario. El cálculo del coeficiente se hace con la siguiente instrucción:

```
carrera[i].coefi+=(hayarea*10/tot)*2;
```

Finalmente, los resultados quedan almacenados en el arreglo *carrera*, el cual es ordenado por coeficiente en orden descendente. Además, se crea el archivo *MEMORIA1.DAT*, el cual contiene los nombres de las carreras, el lugar que ocupa dentro del archivo CARRERAS.DAT y el coeficiente obtenido.

De la misma forma que en el proceso anterior, aquí se emplea la *búsqueda guiada por objetivos o encadenamiento hacia atrás*, ya que se examina una por una cada carrera de acuerdo con los atributos que la definen y de cuerdo con los datos del usuario.

El archivo *SISEXPER.CPP* es quien contiene la función *main*, la cual se encarga de la ejecución del programa. El proceso del sistema experto se divide en las funciones *proceso1*, *proceso2*, *proceso3* y *proceso4*, las cuales son ejecutadas por *main*.

LA GUIA DE CARRERAS

Además del sistema experto, nuestro sistema cuenta con una guía de carreras del estado de México. La figura 3.18 muestra el menú de la guía de carreras.

Como se muestra, las carreras se agrupan de la siguiente forma:

a) Ciencias de la Salud.
b) Ciencias Aplicadas.
c) Administración, Finanzas y Mercadotecnia.
d) Ciencias Políticas y Sociales.
e) Ciencias de la Construcción y Recursos Naturales.
f) Ciencias Humanas.
g) Ciencias Puras.
h) Ciencias de la Educación.

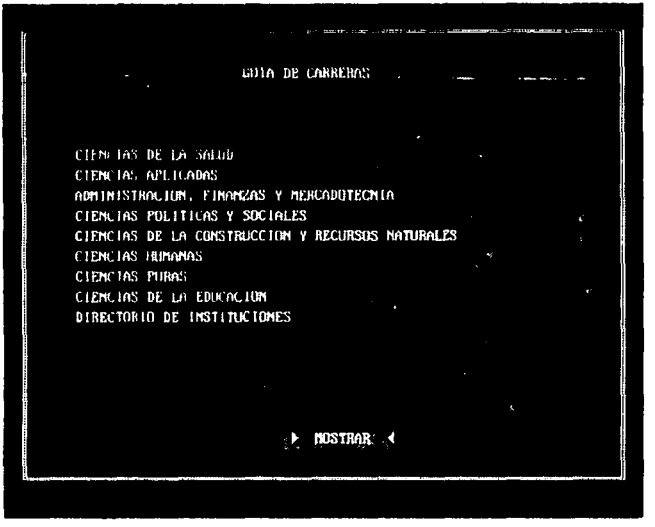

Fig. 3.18. Menú de la guía de carreras.

Al elegir alguna de las opciones, aparecerá una lista de las carreras contenidas en ese grupo. Enseguida podemos seleccionar una de ellas y observar su plan de estudio y duración en semestres como se muestra en la figura 3.19.

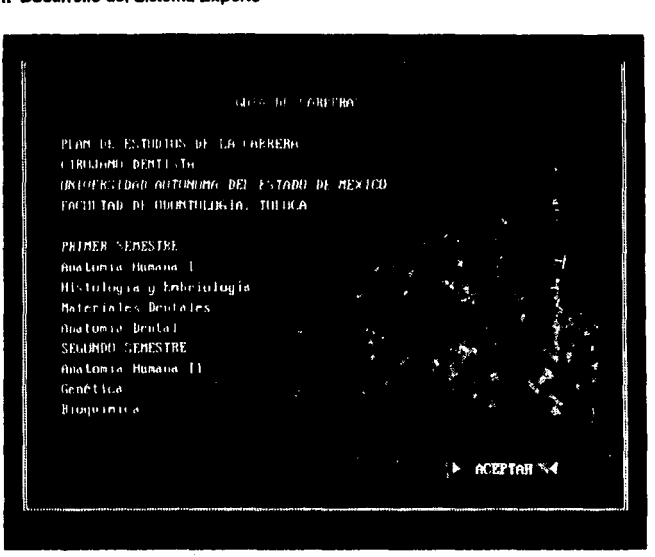

Fig. 3.19. Plan de estudios de una carrera seleccionada.

Además se cuenta con un directorio de instituciones donde se pueden consultar las direcciones y teléfonos de las mismas.

La información de los planes de estudio está almacenada por áreas en los siguientes archivos:

GUIA2.TXT - Ciencias de la Salud.
GUIA3.TXT - Ciencias Aplicadas.
GUIA4.TXT - Administración, Finanzas y Mercadotecnia.
GUIA5.TXT - Ciencias Políticas y Sociales.
GUIA6.TXT - Ciencias de la Construcción y Recursos Naturales.
GUIA7.TXT - Ciencias Humanas.
GUIA8.TXT - Ciencias Puras.
GUIA9.TXT - Ciencias de la Educación.

El Directorio de Instituciones se almacena en *DIRECT.TXT.*

Las funciones que manipulan la guía de carreras se encuentran en el archivo *GUIA.CPP.* El acceso a los planes de estudio se hace por medio de otros archivos que contienen el nombre de la carrera, el archivo que contiene su plan de estudios, el número de línea de dicho archivo en donde comienza su descripción y el número de línea donde termina. Por ejemplo, el archivo *GCARR4.DAT,* contiene la descripción de la carrera *Informática y Computación* de la siguiente manera::

INFORMATICA Y COMPUTACION (U.T. NEZAHUALCOYOTL)
GUIA3.TXT
3455
3518

Lo cual nos indica que su plan de estudios está contenido a partir de la línea 3455 y hasta la 3518 en el archivo GUIA3.TXT.

Esta forma de manejar la información es conocida por algunos autores como *archivos encadenados.* A continuación se muestra como están encadenados nuestros archivos:

Archivo	**Contiene información sobre...**
GCARR2.DAT	GUIA2.TXT
GCARR4.DAT	GUIA3.TXT
GCARR6.DAT	GUIA4.TXT
GCARR8.DAT	GUIA5.TXT
GCARR10.DAT	GUIA6.TXT
GCARR12.DAT	GUIA7.TXT
GCARR14.DAT	GUIA8.TXT
GCARR16.DAT	GUIA9.TXT
GCARR18.DAT	DIRECT.TXT

De esta forma, cuando se requiera actualizar los planes de estudio, tendrán que ser modificados los archivos que contienen los mismos, y aquellos que tienen su descripción.

Los archivos que contienen los planes de estudio y el Directorio de Instituciones, son archivos ASCII por lo que pueden ser visualizados con cualquier editor de textos o procesador de palabras.

REQUERIMIENTOS DEL SISTEMA

Para correr nuestro sistema se requiere de una computadora personal con ratón y se necesita 1.4 Mbytes de memoria en disco. El sistema maneja colores por lo que es preferible tener un monitor a color. Si se quiere imprimir los resultados, es obvio que se necesitará una impresora.

III.9

Comprobación y Validación del Sistema

Una vez desarrollado el sistema, la siguiente etapa es la comprobación de que el sistema funciona. Como ya hemos comentado, el equipo de especialistas para el desarrollo de nuestro sistema experto fue el *Departamento de Orientación Vocacional de la Escuela Nacional Preparatoria plantel Ezequiel A. Chávez (7).* Dado lo anterior, la etapa de validación fue también desarrollada con la ayuda de los orientadores y alumnos de esta institución educativa.

La etapa de validación se logró mediante la aplicación del sistema experto a 10 estudiantes de quinto año de la *Preparatoria 7*, uno de los cuales fue observado por dos orientadores vocacionales.

El tiempo que se tardaron los estudiantes en contestar las preguntas, analizar los resultados obtenidos por el sistema experto y consultar la guía de carreras fue de 50 minutos en promedio. Todos coincidieron en que el sistema es fácil de manejar y que cuenta con tal cantidad de información que lo hace muy completo. Dijeron también que el sistema de ayuda es bueno y que es muy interesante la información que contiene. Comentaron que es más interesante contestar los cuestionarios por medio del sistema y enseguida recibir resultados que realizarlo en papel y hacer todo el proceso para recibir resultados. En general, dijeron que les ayudó bastante en el conocimiento de sí mismos; todos, excepto una persona, estuvieron de acuerdo en los resultados que les mostró el sistema experto.

Por otra parte, los dos orientadores comentaron que nuestro sistema es una buena herramienta para mostrar al alumno un panorama general y enseñarle los diversos factores que deben tomarse en cuenta en la elección de profesión. Dijeron que mientras que con papel y lápiz el aplicar al alumno los dos primeros cuestionarios del sistema y obtener los resultados (perfil de interés-habilidad) les toma en promedio cuatro horas (cuatro sesiones de 50 minutos), el sistema experto lo hace en 30 minutos. Dijeron que este era un factor muy importante, ya que a la semana se cuenta con solo una sesión de orientación vocacional para

cada grupo, por lo que se tiene una discontinuidad en el proceso al dejarlo parcialmente incompleto cada semana. Dijeron que el número de preguntas hechas a los alumnos es el adecuado, no siendo demasiado cansado para ellos. Coincidieron también en que el sistema es muy completo y maneja una buena cantidad de información.

Al preguntarles como emplearían este sistema en su proceso de orientación a los alumnos, dijeron que sería una excelente herramienta, la cual sería aplicada después de dar una plática de información global y que una vez obtenidos los resultados por el sistema experto, se tratarían dudas más específicas por parte de los estudiantes. Dijeron que el sistema sirve para despertar la inquietud en el alumno de manera que le encamina a la investigación y que para esto es también de gran ayuda la información contenida en la guía de carreras. De acuerdo con sus opiniones, el sistema experto les proporciona a los orientadores un punto de partida en el proceso de orientación vocacional.

Gracias a los comentarios de los orientadores y alumnos, fue posible corregir algunas fallas del sistema.

En general, quedamos satisfechos con las opiniones de los usuarios de nuestro sistema, por lo cual seguimos adelante en el desarrollo de esta tesis y teniendo como un proyecto firme el que nuestro sistema experto sea utilizado al menos por la *Escuela Nacional Preparatoria plantel Ezequiel A. Chávez (7).*

III.10

Mantenimiento del Sistema

La parte de nuestro sistema que más necesitada está de mantenimiento es la guía de carreras, ya que los planes de estudio constantemente se están actualizando. Cuando haya que actualizar algún plan de estudios, se deberán modificar los archivos correspondientes (*gcarr?.dat y guia?.txt*), sin tener que modificar algún programa. De igual forma se hará cuando se quiera agregar o quitar alguna carrera.

En caso de que alguna vez se quisiera agregar o eliminar algún cuestionario del sistema experto, esto si requerirá la modificación de los programas.

Conclusiones

Ciertamente el proceso de construcción del sistema *Doctor Vocacional* fue un trabajo que requirió de tiempo y dedicación. El sistema tardó en desarrollarse aproximadamente un año y medio, tiempo durante el cual se hicieron entrevistas e investigaciones, se realizaron programas de computadora y archivos de información y se tuvo una etapa de comprobación y validación del sistema.

Antes de comenzar la construcción del sistema, investigamos toda la teoría que sustenta este proyecto, es decir, los capítulos I y II. Se tuvo una buena documentación acerca de la Teoría de los Sistemas Expertos, lo cual permitió encontrar un método para la construcción del nuestro y además nuestras investigaciones y entrevistas acerca de la Orientación Vocacional permitieron que nuestro sistema tuviera, a nuestro parecer, una buena fundamentación para formar su base de conocimiento.

Durante la etapa de validación y comprobación pudimos darnos cuenta que el trabajo logrado es en sí aceptable y funcional y nos dio gusto el que en un futuro no lejano éste pueda solucionar problemas reales. De acuerdo con nuestro análisis costo-beneficio, pudimos darnos cuenta también que los beneficios son mucho mayores a los costos.

Podemos decir que se cumplieron los dos objetivos que principalmente se perseguían al realizar este trabajo de tesis: primero, realizar un proyecto en donde se aplicaran los conocimientos ya tenidos y se adquirieran nuevos y segundo, lograr una aplicación funcional con aplicación real.

Por lo que respecta a la Inteligencia Artificial y a los Sistemas Expertos, creemos que esta es un área de la computación con aplicación actual y con una amplia perspectiva de desarrollo en el futuro. Hablar de sistemas expertos hoy en día es más común que hace algunos años. En cuanto a sistemas expertos con aplicación en la Orientación Vocacional, existen ya instituciones educativas que los emplean y entre ellas se encuentra la Dirección General de Orientación Vocacional de nuestra Universidad Nacional Autónoma de México.

Para concluir, me siento satisfecho con el trabajo realizado y aunque un proyecto siempre puede mejorarse creo que las bases están puestas para alcanzar el éxito en la implantación de nuestro sistema: **El Doctor Vocacional**.

Apéndice A

Listado de los Programas

ARCHIVO DRVOC.CPP.- Contiene la función main(), la cual controla todo el programa. Ejecuta el menú de inicio, en el cual se elige ejecutar el sistema experto, ejecutar la guia de carreras o bien salir del sistema.

```
#include <c:\tesis\marco.h>

// Funciones contenidas en otros archivos

int guia();          // contenido en <guia.cpp>
int sistema_experto(); // contenido en <sisexper.cpp>
int mainimagen();     // contenido en imagen.cpp


int menufinal();

int menufinal()
// Despliega un men£ el cual contiene las opciones: ejecutar el sistema
// experto, ejecutar la guia de carreras y salir del sistema.
{
  unsigned char *p;
  register int i,j;
  int xpos;
  char car;
  char opcion[2][26];
  char continua;
  mouse raton;
  _setcursortype(_NOCURSOR);
  pantalla a1(5,2,75,23,15,5,0,1,"      ");
  botton b1(50,21,65,22,1,3,0,5,"TERMINAR");
  botton e1(9,6,14,7,1,3,0,5," 1 ");
  botton e2(9,14,14,15,1,3,0,5," 2 ");
  a1.pon_fondo(1,3);
  a1.pon_marco();
  b1.pon();
  e1.pon();
  e2.pon();
  e1.ilumina(3,15);
  a1.escribe(1,1,15,3,"    DR. VOCACIONAL                    | VER. 1.0   ");
  strcpy(opcion[0],"SISTEMA EXPERTO");
  strcpy(opcion[1],"GUIA DE CARRERAS");
 a1.escribe(18,7,11,5,opcion[0]);
 a1.escribe(18,15,15,5,opcion[1]);
 a1.lhor(40,5,70,0,0);
 a1.lhor(40,18,70,15,0);
 a1.lver(40,5,18,0,0);
 a1.lver(70,5,18,15,0);
 a1.escribe(40,5,0,5,"Ú");
 a1.escribe(70,5,0,5,"¿");
 a1.escribe(40,18,0,5,"À");
 a1.escribe(70,18,15,5,"Ù");
 a1.escribe(45,12,11,5,"   CONSULTA  DEL ");
 a1.escribe(45,13,11,5,"  SISTEMA EXPERTO EN  ");
 a1.escribe(45,14,11,5,"ORIENTACION VOCACIONAL");
 raton.inicia();
```

```
raton.activa();
j=raton.x;
i=11;
do {
  raton.activa();
  a1.escribe(47,7,i,5,"DOCTOR VOCACIONAL");
  a1.escribe(49,8,i,5," versión 1.0 ");
  a1.escribe(49,9,i,5,"  por gcml ");
  if(i==11)
    i=15;
  else
    i=11;
  delay(100);
} while(!kbhit()&&(j==raton.x));
a1.escribe(47,7,14,5,"DOCTOR VOCACIONAL");
a1.escribe(49,8,14,5," versión 1.0 ");
a1.escribe(49,9,14,5,"  por gcml ");
a1.escribe(45,12,11,5,"  CONSULTA  DEL ");
a1.escribe(45,13,11,5,"  SISTEMA EXPERTO EN ");
a1.escribe(45,14,11,5,"ORIENTACION VOCACIONAL");
continua='1';
xpos=1;
do
{
  raton.activa();
  if(kbhit())
  {
    car=leetecla();
    if(car=='E')  // enter
    {
      switch(xpos)
      {
        case 1:
              {
                e1.push(3,15);
                delay(100);
                e1.pon();
          e1.ilumina(3,15);
                raton.desactiva();
          p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
          salva_pant(0,0,79,24,(char *) p);
          // SISTEMA EXPERTO
          sistema_experto();
          restaura_pant(0,0,79,24,(char *) p);
          free(p);
          raton.inicia();
              }
          break;
        case 2:
              {
                e2.push(3,15);
                delay(100);
                e2.pon();
          e2.ilumina(3,15);
                raton.desactiva();
          p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
```

```
            salva_pant(0,0,79,24,(char *) p);
            // GUIA DE CARRERAS
            guia();
            restaura_pant(0,0,79,24,(char *) p);
            free(p);
            raton.inicia();
                }
            break;
        case 3:
                {
               // TERMINAR
                  b1.push(3,15);
                  delay(100);
                  b1.pon();
                  raton.desactiva();
                  return(1);
                }
             break;
    }
}
if(car=='C')  // derecha
{
   if(xpos==2)
   {
     e2.pon();
     a1.escribe(18,15,15,5,opcion[1]);
     a1.escribe(45,12,11,5,"                    ");
     a1.escribe(45,13,11,5," SALIR DEL SISTEMA  ");
     a1.escribe(45,14,11,5,"                    ");
     b1.ilumina(3,15);
     xpos=3;
   }
}
if(car=='D')  // izquierda
{
   if(xpos==3)
   {
     b1.pon();
     e2.ilumina(3,15);
     a1.escribe(18,15,11,5,opcion[1]);
     a1.escribe(45,12,11,5,"  CONSULTA DE LA");
     a1.escribe(45,13,11,5,"GUIA DE CARRERAS EN EL");
     a1.escribe(45,14,11,5,"  ESTADO DE MEXICO  ");
     xpos=2;
   }
}
if(car=='A') // arriba
{
   switch(xpos)
   {
     case(2):
          {
            e2.pon();
            e1.ilumina(3,15);
            a1.escribe(18,7,11,5,opcion[0]);
            a1.escribe(18,15,15,5,opcion[1]);
```

```
                a1.escribe(45,12,11,5,"  CONSULTA  DEL ");
                a1.escribe(45,13,11,5," SISTEMA EXPERTO EN  ");
                a1.escribe(45,14,11,5,"ORIENTACION VOCACIONAL");
                xpos--;
               }
               break;
     }
    }
    if(car=='B')  // abajo
    {
      switch(xpos)
      {
        case(1):
               {
                e1.pon();
                e2.ilumina(3,15);
                a1.escribe(18,15,11,5,opcion[1]);
                a1.escribe(18,7,15,5,opcion[0]);
                a1.escribe(45,12,11,5,"  CONSULTA DE LA");
                a1.escribe(45,13,11,5,"GUIA DE CARRERAS EN EL");
                a1.escribe(45,14,11,5,"  ESTADO DE MEXICO  ");
                xpos++;
               }
               break;
     }
    }
  }
  if(raton.boton==1)
  {
    if((raton.x > 58)&&(raton.x<75)&&(raton.y>21)&&(raton.y<23))   // terminar
    {
      a1.escribe(18,15,15,5,opcion[1]);
      a1.escribe(18,7,15,5,opcion[0]);
      e1.pon();
      e1.pon();
      b1.push(3,15);
      while(raton.boton==1)
       raton.activa();
      b1.pon();
      raton.desactiva();
      return(1);
    }
    if((raton.x > 8)&&(raton.x<15)&&(raton.y>6)&&(raton.y<8)) // sistema experto
    {
      b1.pon();e2.pon();
      a1.escribe(18,15,15,5,opcion[1]);
      a1.escribe(18,7,11,5,opcion[0]);
      e1.ilumina(3,15);
      xpos=1;
      e1.push(3,15);
      while(raton.boton==1)
      raton.activa();
      raton.desactiva();
      e1.pon();
      e1.ilumina(3,15);
      p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
```

```
        salva_pant(0,0,79,24,(char *) p);
        // SISTEMA EXPERTO
        sistema_experto();
        restaura_pant(0,0,79,24,(char *) p);
        free(p);
        raton.inicia();
      }
      if((raton.x > 8)&&(raton.x<15)&&(raton.y>14)&&(raton.y<16)) //guia de carreras
      {
        b1.pon();e1.pon();
        a1.escribe(18,15,11,5,opcion[1]);
        a1.escribe(18,7,15,5,opcion[0]);
        a1.escribe(45,12,11,5,"  CONSULTA DE LA");
        a1.escribe(45,13,11,5,"GUIA DE CARRERAS EN EL");
        a1.escribe(45,14,11,5,"  ESTADO DE MEXICO  ");
        e2.ilumina(3,15);
        xpos=2;
        e2.push(3,15);
        while(raton.boton==1)
          raton.activa();
        raton.desactiva();
        e2.pon();
        e2.ilumina(3,15);
        p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
        salva_pant(0,0,79,24,(char *) p);
        // GUIA DE CARRERAS
        guia();
        restaura_pant(0,0,79,24,(char *) p);
        free(p);
        raton.inicia();
      }
    }
  } while(continua=='1');
  return(0);
}


int main()
{
  mainimagen();
  menufinal();
  textcolor(7);
  textbackground(0);
  clrscr();
  _setcursortype(_NORMALCURSOR);
  return(1);
}
```

```
// ARCHIVO IMAGEN.CPP.- Es parte de la interface con el usuario
// y sus funciones se encargan de desplegar la portada de inicio.
```

```
/* Programa que lee imagenes en formato PCX */

#include <alloc.h>
#include <graphics.h>
#include <mem.h>
#include <conio.h>
#include <c:\tesis\marco.h>

void pon_imagen(char *archi[]);  // pone una imagen formato pcx
ReadPcxLine(char far *p, FILE *fp);  // lee lineas del archivo
void Init();    // Activa modo grafico */
void deinit();  // Desactiva modo grafico */
void UnpackPcxFile(FILE *fp);  // controla la lectura del archivo
void setvgapalette(char *p);  // pone la paleta de colores
void errorimagen(char *s);    // detecta errores en la imagen

typedef struct {
  char manufacturer;
  char version;
  char encoding;
  char bits_per_pixel;
  int  xmin, ymin;
  int  xmax, ymax;
  int  hres;
  int  vres;
  char palette[48];
  char reserved;
  char colour_planes;
  int  bytes_per_line;
  int  palette_type;
  char filler[58];
  } PCXHEAD;

PCXHEAD header;
unsigned int width, depth;
unsigned int bytes, bits;
char palette[768];



void pon_imagen(char archi[])
{
  FILE *fp;

// if(argc<=1) errorimagen("Se necesita el path del archivo");

  if((fp=fopen(archi,"rb")) ==NULL) errorimagen("Error al abrir el archivo");

  if(fread((char *)&header,1,sizeof(PCXHEAD),fp) != sizeof(PCXHEAD))
    errorimagen("Error al leer la cabecera");
```

```
 if(header.manufacturer != 0x0a || header.version != 5)
   errorimagen("Este no es un archivo manejable");

 if(header.bits_per_pixel == 1) bits=header.colour_planes;
   else bits=header.bits_per_pixel;

 if(bits != 8) errorimagen ("Este archivo tiene un numero de colores equivocado");

 if(!fseek(fp,-769L, SEEK_END)) {
   if (fgetc(fp) != 0x0c || fread(palette,1,768,fp) != 768)
     errorimagen("Error al leer la paleta");
 } else errorimagen("Error al posicionarse sobre la paleta");

 fseek(fp,128L,SEEK_SET);

 width = (header.xmax-header.xmin)+1;
 depth = (header.ymax-header.ymin)+1;
 bytes = header.bytes_per_line;

 UnpackPcxFile(fp);
 fclose(fp);
}



void  UnpackPcxFile(FILE *fp)
{
  int i;
  unsigned char far *pant;
/*  int ch; */
  init();
  setvgapalette(palette);
  for(i=0;i<depth;++i) {
    if(i>=200) break;
    pant = (char far *)MK_FP(0xa000,i*320);
    ReadPcxLine(pant,fp);
    }
  sound(5000);
  delay(50);
  sound(25);
  delay(50);
  nosound();
  getch();
  sound(5000);
  delay(50);
  sound(25);
  delay(50);
  nosound();
  deinit();
}
```

```
ReadPcxLine(char far *p, FILE *fp)
{
  int n=0,c,i;
  do {
     c=fgetc(fp) & 0xff;
     if((c & 0xc0) == 0xc0) {
          i=c & 0x3f;
          c=fgetc(fp);
          while(i--) p[n++]=c;
          }
     else p[n++]=c;
     } while(n<bytes);
     return(n);
}



void  init()    /* Activa modo grafico */
{
  union REGS r;
  r.x.ax=0x0013;
  int86(0x10,&r,&r);
}



void  deinit()   /* Desactiva modo grafico */
{
  union REGS r;
  r.x.ax=0x0003;
  int86(0x10,&r,&r);
 }



void  setvgapalette(char *p)
{
  union REGS r;
  struct SREGS sr;
  int i;

  for(i=0;i<768;++i) p[i]=p[i]>>2;
  r.x.ax=0x1012;
  r.x.bx=0;
  r.x.cx=256;
  r.x.dx=FP_OFF(p);
  sr.es=FP_SEG(p);
  int86x(0x10,&r,&r,&sr);
}

void  errorimagen(char *s)
 {
  error(1,"INICIO.PCX");
//  puts(s);
  exit(1);
```

```
}

int mainimagen()
{
  pon_imagen("INICIO.PCX");
  return(1);
}
```

```
// ARCHIVO SISEXPER.CPP.- Es el motor de inferencia del  sistema
// experto. Ademas de obtener resultados,   coordina la   aplicacion
// de los cuestionarios.
```

```
// coordina los cuestionarios de intereses, habilidades y valores
// obtiene los tres mejores pares de interes-habilidad
// obtiene la personalidad de la persona
// es parte del MOTOR DE INFERENCIA

#include "c:\tesis\marco.h"

//FUNCIONES CONTENIDAS EN OTROS ARCHIVOS Y USADAS POR EL PRESENTE

int main_interes();     // <cuestion.cpp>
   // coordina el cuestionario de intereses
int resulta_int();      // <cuestion.cpp>
   // muestra los resultados del perfil de intereses
int main_habilidad();   // <cues1.cpp>
   // coordina la aplicacion del cuestionario de habilidades
int resulta_habi();     // <cues1.cpp>
  // realiza los resultados del perfil de habilidades
int main_valor();       // <cues2.cpp>
   // coordina el cuestionario de valores
int main_ocupa();       // <cues3.cpp>
   // coordina el cuestionario de ocupaciones
int main_area();        // <cues4.cpp>
   // coordina el cuestionario de areas de ocupacion
int mainfinal(char mejorper[4]);       // <final.cpp>
   // coordina el menu de resultados finales
int main_resulta();     // <resulta.cpp>
   // muestra introducciones a las primeras inferencias
int pon_mejorih(char mejorih[4]);  // <resulta.cpp>
   // muestra las tres mejores parejas interes-habilidad
int grafica_inter_y_habi();        // <resulta.cpp>
   // grafica el perfil de interes-habilidad
int graficaper(int valorper[6]);   // <resulta.cpp>
   // grafica el perfil de personalidad
int main_intro();     // <intro.cpp>
   // coordina la introduccion inicial
int imprime(int valorper[6]);   // <imprime.cpp>
   // imprime el reporte de resultados
int datos();  // datos.cpp
   // captura los datos personales


// FUNCIONES CONTENIDAS EN ESTE ARCHIVO
int ordena(char mejorih[4]);
  // regresa las tres mejores parejas interes-habilidad
int obten_persona(char mejorih[4], int valorper[6]);
  // obtiene los tres mejores tipos de personalidad
```

```
int ordenaper(int valorper[6], char mejorper[4]);
  // ordena el arreglo de personalidad
int motor1(char mejorper[4]);
  // Obtiene los coeficientes para cada una de las 449 carreras
  // contempladas, tomando en cuenta la personalidad, rama y campo
  // de trabajo y area de estudio.
// El proceso del sistema experto se divide en
// proceso1, proceso2, proceso3 y proceso4
int proceso1(char mejorih[4]);
int proceso2(char mejorih[4], int valorper[6], char mejorper[4]);
int proceso3();
int proceso4(char mejorper[4]);
int menufinal(char mejorih[4], char mejorper[4], int valorper[6]);
  // coordina el menu final de resultados




int ordena(char mejorih[4])
// regresa las tres mejores parejas interes-habilidad
{
  FILE *fp;
  int evinter[10];
  int evhabi[10];
  register int i,j, pasa;
  typedef struct
  {
    char elemento[3];
    int puntos;
  } arreglo;
  arreglo c;
  arreglo intereses[10];
  arreglo habilidades[10];
  // lee evinter del archivo memoria.dat que es la memoria de trabajo
  if((fp=fopen("memoria.dat","r"))==NULL)
  {
    error(1,"MEMORIA.DAT");
    return(0);
  }
  for(i=0;i<10;i++)
    fscanf(fp,"%d ",&evinter[i]);
  fscanf(fp,"\n");
  // lee evhabi del archivo memoria.dat
  for(i=0;i<10;i++)
    fscanf(fp,"%d ",&evhabi[i]);
  fscanf(fp,"\n");
  fclose(fp);
  // vacia evhabi y evinter en el nuevo arreglo
  for(i=0;i<10;i++)
  {
    intereses[i].puntos=evinter[i];
    habilidades[i].puntos=evhabi[i];
  }
  // vaciar la clave de intereses y habilidades
```

```
strcpy(intereses[0].elemento,"I0"); strcpy(habilidades[0].elemento,"H0");
strcpy(intereses[1].elemento,"I1"); strcpy(habilidades[1].elemento,"H1");
strcpy(intereses[2].elemento,"I2"); strcpy(habilidades[2].elemento,"H2");
strcpy(intereses[3].elemento,"I3"); strcpy(habilidades[3].elemento,"H3");
strcpy(intereses[4].elemento,"I4"); strcpy(habilidades[4].elemento,"H4");
strcpy(intereses[5].elemento,"I5"); strcpy(habilidades[5].elemento,"H5");
strcpy(intereses[6].elemento,"I6"); strcpy(habilidades[6].elemento,"H6");
strcpy(intereses[7].elemento,"I7"); strcpy(habilidades[7].elemento,"H7");
strcpy(intereses[8].elemento,"I8"); strcpy(habilidades[8].elemento,"H8");
strcpy(intereses[9].elemento,"I9"); strcpy(habilidades[9].elemento,"H9");
// ordenar el arreglo de intereses, metodo de la burbuja
for(i=0;i<10;i++)
  for(j=9;j>i;j--)
        if (intereses[j-1].puntos < intereses[j].puntos)
        {
          c=intereses[j-1];
          intereses[j-1]=intereses[j];
          intereses[j]=c;
        }
 // ordenar el arreglo de habilidades
for(i=0;i<10;i++)
  for(j=9;j>i;j--)
        if (habilidades[j-1].puntos < habilidades[j].puntos)
        {
          c=habilidades[j-1];
          habilidades[j-1]=habilidades[j];
          habilidades[j]=c;
        }
// obtener las tres mejores parejas habilidad-interes mejorhi
// queda almacenado solo el numero del interes y habilidad
i=0;
if(intereses[0].elemento[1]==habilidades[0].elemento[1])
{
  mejorih[i]=intereses[0].elemento[1];  // mejor interes y mejor habilidad
  i++;
  pasa=1;
}
if((intereses[1].elemento[1]==habilidades[0].elemento[1])
   ||(intereses[1].elemento[1]==habilidades[1].elemento[1]))
{
  mejorih[i]=intereses[1].elemento[1];  //2o mejor interes y mejor habilidad
  i++;                                  // o 2o interes y 2a habilidad
}
if(i==2)
{
  mejorih[i]=intereses[2].elemento[1];  // el tercer mejor interes
  i++;
}
if(i==1)
{
  if(pasa==1)   // ya selecciono el mejor interes
  {
        mejorih[i]=intereses[1].elemento[1];   //2o mejor interes
        i++;
  }
  else
```

```
    {
        mejorih[i]=intereses[0].elemento[1];  //mejor interes
        i++;
    }
    if(pasa==1)
    {
        mejorih[i]=intereses[2].elemento[1];   // tercer mejor interes
        i++;
    }
    else
    {
        mejorih[i]=habilidades[0].elemento[1];  // mejor habilidad
        i++;
    }
  }
  if(i==0)
  {
    mejorih[i]=intereses[0].elemento[1];   //mejor interes
    i++;
    mejorih[i]=intereses[1].elemento[1];   //2o mejor interes
    i++;
    mejorih[i]=habilidades[0].elemento[1];  // la mejor habilidad
    i++;
  }
  mejorih[3]='\0';
  return(1);
}




int obten_persona(char mejorih[4], int valorper[6])
// obtiene el coeficiente para cada tipo de personalidad de acuerdo
// a los datos obtenidos
{
  char c;
  FILE *fp;
  float coefi;
  int estan, encuentra;
  register int l,j,k, total;
  char valor[22];
  char linea[41];
  typedef struct
  {
    char per[14];
    int coef;
  } personalidad;
  personalidad persona[6], aux;
  // lee valor del archivo memoria.dat que es la memoria de trabajo
  if((fp=fopen("memoria.dat","r"))==NULL)
  {
    error(1,"MEMORIA.DAT");
```

```
  return(0);
}
valor[21]='\0';
// saltar los dos primeros renglones
for(i=0;i<2;i++)
  do
       fscanf(fp,"%c",&c);
  while(c!='\n');
for(i=0;i<22;i++)
  fscanf(fp,"%c",&valor[i]);
fscanf(fp,"\n");
fclose(fp);
// abrir el archivo base1.dat que contiene la descripcion de cada
// personalidad
if((fp=fopen("base1.dat","r"))==NULL)  {
  error(1,"BASE1.DAT");
  return(0);
}
// para cada personalidad calcular su coeficiente
// saltar la primera linea
for(i=1;i<2;i++)     // la primera linea no se lee
  do {
       fscanf(fp,"%c",&c);
  } while(c!='\n');
for(j=1;j<7;j++)          // el archivo tiene un total de 7 lineas
{
  total=0;
  // limpiar el arreglo
  for(i=0;i<41;i++)
  linea[i]=' ';
  linea[40]='\0';
  for(i=1;i<=41;i++)
  {
       fscanf(fp,"%c",&c);
       if((i>13)&&(c!='\n'))
         if(c!=' ')
           total++;
       if(c!='\n')
         linea[i-1]=c;
       else
         i=41;
  }
  estan=0;   // registra las condiciones cumplidas
  // calcular el coeficiente como estan/total
  // calculo de habilidad - interes
  for(i=16;i<21;i++)
       if(linea[i]!=' ')
       {
         // checar si esta en memoria de trabajo
         encuentra=0;    // indicador de presencia en memoria  de trab.
         k=0;
         while((k<3)&&(encuentra==0))
         {
           if(mejorih[k]==linea[i])
                encuentra=1;
           else
```

```
                k++;
            }
            if(encuentra==1)
            {
              if(k==0)      // si es el primer mejorih lo toma como doble
                  estan++;
              estan++;
            }
          }
    // calculo de valores
    for(i=31;i<40;i++)
    {
          if(linea[i]!=' ')
          {
            switch(linea[i])
            {
              case '0': k=0; break;
              case '1': k=1; break;
              case '2': k=2; break;
              case '3': k=3; break;
              case '4': k=4; break;
              case '5': k=5; break;
              case '6': k=6; break;
              case '7': k=7; break;
              case '8': k=8; break;
              case '9': k=9; break;
              case 'A': k=10; break;
              case 'B': k=11; break;
              case 'C': k=12; break;
              case 'D': k=13; break;
              case 'E': k=14; break;
              case 'F': k=15; break;
              case 'G': k=16; break;
              case 'H': k=17; break;
              case 'I': k=18; break;
              case 'J': k=19; break;
              case 'K': k=20; break;
            }
            if(valor[k]=='p')
              estan++;
          }
    }
    coefi=estan*100/(total+1);
    // vaciar los resultados en el arreglo persona
    for(i=0;i<13;i++)
          persona[j-1].per[i]=linea[i];
    persona[j-1].per[13]='\0';
    persona[j-1].coef=coefi;
  }
  fclose(fp);
  // vaciar los coeficientes en el arreglo valorper
  for(i=0;i<6;i++)
    valorper[i]=persona[i].coef;
  return(1);
}
```

```
int ordenaper(int valorper[6], char mejorper[4])
{
  unsigned char *p;
  pantemerge a1(15,4,65,21,15,7,0,8,"ayuper.txt");
  char mejor[3][14];
  mouse raton;
  int xpos;
  char continua, car;
  pantalla d1(2,1,77,23,15,5,3,1,"res31.txt");
  botton e2(35,21,50,22,1,3,0,5,"ACEPTAR");
  botton e1(55,21,70,22,1,3,0,5,"AYUDA");
  d1.pon_fondo(1,3);
  d1.pon_marco();
  d1.pon_texto(13,7,1,17,15);
  e2.pon();
  e1.pon();
  e2.ilumina(3,15);
  d1.escribe(28,4,14,5,"PERFIL DE PERSONALIDAD");
  _setcursortype(_NOCURSOR);
  d1.lhor(26,5,52,1,1);
  d1.lver(52,4,4,1,1);
  register int i,j;
  typedef struct
  {
    char per;
    int coef;
  } personalidad;
  personalidad persona[6], aux;
  // vaciar los datos en el arreglo persona
  for(i=0;i<6;i++)
  {
    persona[i].coef=valorper[i];
    switch(i)
    {
          case 0: persona[i].per='R'; break;
          case 1: persona[i].per='I'; break;
          case 2: persona[i].per='A'; break;
          case 3: persona[i].per='S'; break;
          case 4: persona[i].per='E'; break;
          case 5: persona[i].per='C'; break;
    }
  }
  // ordenar por el metodo de la burbuja
  for(i=0;i<6;i++)
    for(j=5;j>i;j--)
          if (persona[j-1].coef < persona[j].coef)
          {
            aux=persona[j-1];
            persona[j-1]=persona[j];
            persona[j]=aux;
          }
  // vaciar las tres mejores personalidades en mejorper
  for(i=0;i<3;i++)
    mejorper[i]=persona[i].per;
  mejorper[3]='\0';
  for(i=0;i<3;i++)
```

```
  switch(mejorper[i])
  {
       case 'R': strcpy(mejor[i],"REALISTA");  break;
       case 'I': strcpy(mejor[i],"INVESTIGATIVO");  break;
       case 'A':;strcpy(mejor[i],"ARTISTICO");   break;
       case 'S': strcpy(mejor[i],"SOCIAL"); break;
       case 'E': strcpy(mejor[i],"EMPRENDEDOR");  break;
       case 'C': strcpy(mejor[i],"CONVENCIONAL");  break;
  }
d1.escribe(30,12,15,5,"1.");
d1.escribe(30,14,15,5,"2.");
d1.escribe(30,16,15,5,"3.");
d1.escribe(33,12,14,5,mejor[0]);
d1.escribe(33,14,14,5,mejor[1]);
d1.escribe(33,16,14,5,mejor[2]);
continua='1';
raton.inicia();
xpos=1;
do
{
  raton.activa();
  if(kbhit())
  {
       car=leetecla();
       if(car=='E')  // enter
       {
         if(xpos==1)              // aceptar
         {
        e1.pon();
           e2.push(3,15);
           delay(100);
           e2.pon();
           raton.desactiva();
           return(1);
         }
         if(xpos==2)              // ayuda
         {
           e1.push(3,15);
           delay(100);
           e1.pon();
        raton.desactiva();
        p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
           salva_pant(0,0,79,24,(char *) p);
        a1.pon_marco();
        a1.control(0);
        restaura_pant(0,0,79,24,(char *) p);
        free(p);
        e2.pon();
        e1.ilumina(3,15);
        raton.inicia();
         }
       }
       if((car=='C')&&(xpos==1))  // derecha
       {
         e2.pon();
         e1.ilumina(3,15);
```

```
            xpos=2;
          }
          if((car=='D')&&(xpos==2))  // izquierda
          {
            e1.pon();
            e2.ilumina(3,15);
            xpos=1;
          }
      }
      if(raton.boton==1)
      {
          if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
          {
          e1.pon();
            e2.push(3,15);
            while(raton.boton==1)
              raton.activa();
            e2.pon();
            raton.desactiva();
            return(1);
          }
          if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // ayuda
          {
            e2.pon();
            e1.push(3,15);
            while(raton.boton==1)
              raton.activa();
            e1.pon();
          raton.desactiva();
          p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
            salva_pant(0,0,79,24,(char *) p);
          a1.pon_marco();
          a1.control(0);
          restaura_pant(0,0,79,24,(char *) p);
          free(p);
          e1.ilumina(3,15);
          raton.inicia();
          xpos=2;
          }
      }
    } while(continua=='1');
    return(1);
}



int motor1(char mejorper[4])
// Obtiene los coeficientes para cada una de las 449 carreras
// contempladas, tomando en cuenta la personalidad, rama y campo
// de trabajo y area de estudio.
{
  typedef struct
  {
    int nocar;
```

```
  int coefi;
} carreras;
carreras carrera [449],aux;
unsigned char *p;
pantalla a1(13,6,63,17,15,7,0,8,"      ");
char c;
int oc, ar, tot, hayocupa, hayarea;
register int i,j,k,x;
char per[4];
char ocupa[59];   // 58 ocupaciones
char area[41];    // 40 areas
FILE *fp,*fc;
char nombre[43];
nombre[42]='\0';
ocupa[58]='\0';
area[40]='\0';
if((fp=fopen("memoria.dat","r"))==NULL)
{
  error(1,"MEMORIA.DAT");
  return(0);
}
// saltar los tres primeros renglones
for(i=0;i<3;i++)
  do
      fscanf(fp,"%c",&c);
  while(c!='\n');
// leer ocupacion
for(i=0;i<59;i++)
  fscanf(fp,"%c",&ocupa[i]);
fscanf(fp,"%c",&c);
// leer areas de memoria de trabajo
for(i=0;i<41;i++)
  fscanf(fp,"%c",&area[i]);
fscanf(fp,"\n");
fclose(fp);
// abrir el archivo carreas.dat
if((fp=fopen("carreras.dat","r"))==NULL)
{
  error(1,"CARRERAS.DAT");
  return(0);
}
//saltar los primeros dos renglones
for(i=0;i<2;i++)
  do
      fscanf(fp,"%c",&c);
  while(c!='\n');
for(i=0;i<449;i++)
{
  carrera[i].coefi=0;
  carrera[i].nocar=i;
  // leer de la columna 43 a la 45 la personalidad de la carrera
  for(j=0;j<42;j++)
    fscanf(fp,"%c",&nombre[j]);
  fscanf(fp,"%c%c%c",&per[0],&per[1],&per[2]);
  per[3]='\0';
  for(j=0;j<3;j++)
```

```
      if((mejorper[j]==per[0])||(mejorper[j]==per[1])||(mejorper[j]==per[2]))
        if(j==0)
            carrera[i].coefi+=40;
        else
            if(j==1)
                  carrera[i].coefi+=30;
             else
                  if(j==2)
                      carrera[i].coefi+=20;
   if(carrera[i].coefi>60)   carrera[i].coefi=60;
   // leer rama y campo desde la columna 47
   fscanf(fp," %d ",&tot);
   hayocupa=0;
   for(j=0;j<tot;j++)
   {
     fscanf(fp,"%d ",&oc);
     if(ocupa[oc]=='p')
       hayocupa++;
   }
   carrera[i].coefi+=(hayocupa*10/tot)*2;
   // leer area
   fscanf(fp,"%d ",&tot);
   hayarea=0;
   for(j=0;j<tot;j++)
   {
     fscanf(fp,"%d%c",&ar,&c);
     if(area[ar-1]=='p')  // ar-1 ya que el arreglo comienza en 0 y las claves en 1
       hayarea++;
   }
   carrera[i].coefi+=(hayarea*10/tot)*2;

  }
  // Ordenar el arreglo carrera por el metodo de la Burbuja
  for(i=0;i<449;i++)
    for(j=448;j>i;j--)
         if (carrera[j-1].coefi < carrera[j].coefi)
         {
            aux=carrera[j-1];
            carrera[j-1]=carrera[j];
            carrera[j]=aux;
         }
  clrscr();
  p = (unsigned char *) malloc((160) * (25));  //asignar memoria para el buffer de video
  salva_pant(0,0,79,24,(char *) p);
  _setcursortype(_NOCURSOR);
  a1.pon_marco();
  a1.escribe(20,9,0,7,"        GENERANDO RESULTADOS ");
  a1.escribe(20,13,15,7,"ÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛ");
  // Escribir las carreras ordenadas junto con su
  // coeficiente en el archivo MEMORIA1.DAT
  if((fc=fopen("memoria1.dat","w"))==NULL)
  {
    error(1,"MEMORIA1.DAT");
    return(0);
  }
  fseek(fp,0,0);
```

```
  x=0;
  j=0;
  for(i=0;i<449;i++)
  {
    // obtener el nombre
    //saltar los primeros dos renglones
    fseek(fp,0,0);
    for(k=0;k<carrera[i].nocar+2;k++)
      do
          fscanf(fp,"%c",&c);
    while(c!='\n');
    for(k=0;k<42;k++)
      fscanf(fp,"%c",&nombre[k]);
    fprintf(fc,"%s %d %d\n",nombre,carrera[i].nocar,carrera[i].coefi);
    j++;
    if((j==12)||(i==444))
    {
      j=0;
      a1.escribe(20+x++,13,9,7,"Ú");
      textcolor(15);
    }
  }
  restaura_pant(0,0,79,24,(char *) p);
  free(p);
  fclose(fp);
  fclose(fc);
  return(1);
}



// Division de la ejecucion del programa
int proceso1(char mejorih[4])
{

 main_intro();
 datos();
 main_interes();
 main_habilidad();
 grafica_inter_y_habi();
 ordena(mejorih);
 pon_mejorih(mejorih);
 main_valor();
 main_resulta();
 return(1);
}


int proceso2(char mejorih[4], int valorper[6], char mejorper[4])
{
 obten_persona(mejorih, valorper);    // es parte del motor de inf.
 graficaper(valorper);
 ordenaper(valorper,mejorper);
 return(1);
}
```

```
int proceso3()
{
 // conocimiento  del mundo laboral
 main_ocupa();
 main_area();
 return(1);
}


int proceso4(char mejorper[4])
{
 motor1(mejorper);
 // resultados finales
 mainfinal(mejorper);
 return(1);
}


int menufinal(char mejorih[4], char mejorper[4], int valorper[6])
{
  char opcion[6][26];
  unsigned char *p;
  register int i,j;
  int xpos;
  char car;
  char continua;
  mouse raton;
  _setcursortype(_NOCURSOR);
  pantalla a1(2,1,77,23,15,5,3,1,"      ");
  botton b1(59,21,74,22,1,3,0,5,"TERMINAR");
  botton e1(9,6,14,7,1,3,0,5,"1");
  botton e2(9,9,14,10,1,3,0,5,"2");
  botton e3(9,12,14,13,1,3,0,5,"3");
  botton e4(9,15,14,16,1,3,0,5,"4");
  botton e5(9,18,14,19,1,3,0,5,"5");
  botton e6(9,21,14,22,1,3,0,5,"6");
  a1.pon_fondo(1,3);
  a1.pon_marco();
  b1.pon();
  e1.pon();
  e2.pon();
  e3.pon();
  e4.pon();
  e5.pon();
  e6.pon();
  e1.ilumina(3,15);
  a1.escribe(30,4,14,5,"   MENU FINAL   ");
  a1.lhor(23,5,55,1,1);
 strcpy(opcion[0],"PERFIL DE INTERESES");
 strcpy(opcion[1],"PERFIL DE HABILIDADES");
 strcpy(opcion[2],"PERFIL HABILIDAD-INTERES");
 strcpy(opcion[3],"PERSONALIDAD");
```

```
strcpy(opcion[4],"RESULTADOS FINALES");
strcpy(opcion[5],"IMPRIMIR REPORTE");
for(i=0;i<6;i++)
{
  a1.escribe(18,7+i*3,15,5,opcion[i]);
}
 continua='1';
 a1.escribe(18,7,11,5,opcion[0]);
 raton.inicia();
 xpos=1;
 do
 {
   raton.activa();
   if(kbhit())
   {
        car=leetecla();
        if(car=='E')  // enter
        {
       switch(xpos)
       {
         case 1:
                {
                  e1.push(3,15);
                  delay(100);
                  e1.pon();
               e1.ilumina(3,15);
                  raton.desactiva();
               p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de
video*/
                    salva_pant(0,0,79,24,(char *) p);
               resulta_int();
                    restaura_pant(0,0,79,24,(char *) p);
                    free(p);
               raton.inicia();
                }
             break;
        case 2:
                {
                  e2.push(3,15);
                  delay(100);
                  e2.pon();
               e2.ilumina(3,15);
                  raton.desactiva();
               p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de
video*/
                    salva_pant(0,0,79,24,(char *) p);
               resulta_habi();
                    restaura_pant(0,0,79,24,(char *) p);
                    free(p);
               raton.inicia();
                }
             break;
        case 3:
                {
                  e3.push(3,15);
                  delay(100);
```

```
                e3.pon();
              e3.ilumina(3,15);
                raton.desactiva();
              p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de
video*/
                    salva_pant(0,0,79,24,(char *) p);
              grafica_inter_y_habi();
              ordena(mejorih);
              pon_mejorih(mejorih);
                    restaura_pant(0,0,79,24,(char *) p);
                    free(p);
              raton.inicia();
                }
              break;
        case 4:
              {
                e4.push(3,15);
                delay(100);
                e4.pon();
              e4.ilumina(3,15);
                raton.desactiva();
              p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de
video*/
                    salva_pant(0,0,79,24,(char *) p);
              graficaper(valorper);
              ordenaper(valorper, mejorper);
                    restaura_pant(0,0,79,24,(char *) p);
                    free(p);
              raton.inicia();
                }
              break;
        case 5:
              {
                e5.push(3,15);
                delay(100);
                e5.pon();
              e5.ilumina(3,15);
                raton.desactiva();
              p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de
video*/
                    salva_pant(0,0,79,24,(char *) p);
              mainfinal(mejorper);
                    restaura_pant(0,0,79,24,(char *) p);
                    free(p);
              raton.inicia();
                }
              break;
        case 6:
              {

                e6.push(3,15);
                delay(100);
                e6.pon();
              e6.ilumina(3,15);
                raton.desactiva();
```

```
            p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de
video*/
                  salva_pant(0,0,79,24,(char *) p);
               // imprimir
               imprime(valorper);
                  restaura_pant(0,0,79,24,(char *) p);
                  free(p);
               raton.inicia();
                 }
               break;
         case 7:
                 {
                   b1.push(3,15);
                   delay(100);
                   b1.pon();
                   raton.desactiva();
                   return(1);
                 }
               break;
      }
       }
       if(car=='C')  // derecha
       {
      if(xpos==6)
      {
          e6.pon();
        a1.escribe(18,7+(5)*3,15,5,"IMPRIMIR REPORTE");
          b1.ilumina(3,15);
          xpos=7;
      }
    }
       if(car=='D')  // izquierda
       {
      if(xpos==7)
      {
          b1.pon();
          e6.ilumina(3,15);
        a1.escribe(18,7+(5)*3,11,5,"IMPRIMIR REPORTE");
          xpos=6;
      }
    }
    if(car=='A') // arriba
    {
      switch(xpos)
      {
        case(2):
              {
               e2.pon();
               e1.ilumina(3,15);
               xpos--;
              }
             break;
        case(3):
              {
               e3.pon();
               e2.ilumina(3,15);
```

```
                xpos--;
               }
               break;
        case(4):
               {
                e4.pon();
                e3.ilumina(3,15);
                xpos--;
               }
               break;
        case(5):
               {
                e5.pon();
                e4.ilumina(3,15);
                xpos--;
               }
               break;
        case(6):
               {
                e6.pon();
                e5.ilumina(3,15);
                xpos--;
               }
               break;
      }
      if((xpos!=5)&&(xpos!=7))
      {
        a1.escribe(18,7+(xpos)*3,15,5,opcion[xpos]);
        a1.escribe(18,7+(xpos-1)*3,11,5,opcion[xpos-1]);
      }
      else
      if(xpos==5)
      {
        a1.escribe(18,7+(5)*3,15,5,"IMPRIMIR REPORTE");
        a1.escribe(18,7+(xpos-1)*3,11,5,opcion[4]);
      }
    }
    if(car=='B')  // abajo
    {
      switch(xpos)
      {
        case(1):
               {
                e1.pon();
                e2.ilumina(3,15);
                xpos++;
               }
               break;
        case(2):
               {
                e2.pon();
                e3.ilumina(3,15);
                xpos++;
               }
               break;
        case(3):
```

```
            {
              e3.pon();
              e4.ilumina(3,15);
              xpos++;
            }
            break;
      case(4):
            {
              e4.pon();
              e5.ilumina(3,15);
              xpos++;
            }
            break;
      case(5):
            {
              e5.pon();
              e6.ilumina(3,15);
              xpos++;
            }
            break;
    }
    if((xpos!=6)&&(xpos!=7))
    {
      a1.escribe(18,7+(xpos-2)*3,15,5,opcion[xpos-2]);
      a1.escribe(18,7+(xpos-1)*3,11,5,opcion[xpos-1]);
    }
    else
    if(xpos==6)
    {
      a1.escribe(18,7+(xpos-1)*3,11,5,"IMPRIMIR REPORTE ");
      a1.escribe(18,7+(xpos-2)*3,15,5,opcion[4]);
    }
  }
}
if(raton.boton==1)
{
    if((raton.x > 58)&&(raton.x<75)&&(raton.y>21)&&(raton.y<23))  // terminar
    {
      b1.push(3,15);
      while(raton.boton==1)
        raton.activa();
      b1.pon();
      raton.desactiva();
      return(1);
    }
  if((raton.x > 8)&&(raton.x<15)&&(raton.y>6)&&(raton.y<8)) // carrera 0
    {
      b1.pon();e2.pon(); e3.pon(); e4.pon(); e5.pon(); e6.pon();
    e1.ilumina(3,15);
    xpos=1;
    for(i=0;i<5;i++)
      a1.escribe(18,7+(i)*3,15,5,opcion[i]);
    a1.escribe(18,22,15,5,"IMPRIMIR REPORTE");
    a1.escribe(18,7+(xpos-1)*3,11,5,opcion[xpos-1]);
      e1.push(3,15);
      while(raton.boton==1)
```

```
        raton.activa();
      raton.desactiva();
    e1.pon();
    e1.ilumina(3,15);
    p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
      salva_pant(0,0,79,24,(char *) p);
    resulta_int();
      restaura_pant(0,0,79,24,(char *) p);
      free(p);
    raton.inicia();
     }
  if((raton.x > 8)&&(raton.x<15)&&(raton.y>9)&&(raton.y<11)) //carrera 2
     {
       b1.pon();e1.pon(); e3.pon(); e4.pon(); e5.pon(); e6.pon();
    e2.ilumina(3,15);
    xpos=2;
    for(i=0;i<5;i++)
      a1.escribe(18,7+(i)*3,15,5,opcion[i]);
    a1.escribe(18,22,15,5,"IMPRIMIR REPORTE");
    a1.escribe(18,7+(xpos-1)*3,11,5,opcion[xpos-1]);
      e2.push(3,15);
      while(raton.boton==1)
        raton.activa();
      raton.desactiva();
    e2.pon();
    e2.ilumina(3,15);
    p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
      salva_pant(0,0,79,24,(char *) p);
    resulta_habi();
      restaura_pant(0,0,79,24,(char *) p);
      free(p);
    raton.inicia();
     }
  if((raton.x > 8)&&(raton.x<15)&&(raton.y>12)&&(raton.y<14))//carrera 3
     {
       b1.pon();e1.pon(); e2.pon(); e4.pon(); e5.pon(); e6.pon();
    e3.ilumina(3,15);
    xpos=3;
    for(i=0;i<6;i++)
      a1.escribe(18,7+(i)*3,15,5,opcion[i]);
    a1.escribe(18,7+(xpos-1)*3,11,5,opcion[xpos-1]);
      e3.push(3,15);
      while(raton.boton==1)
        raton.activa();
      raton.desactiva();
    e3.pon();
    e3.ilumina(3,15);
    p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
      salva_pant(0,0,79,24,(char *) p);
    grafica_inter_y_habi();
    ordena(mejorih);
    pon_mejorih(mejorih);
      restaura_pant(0,0,79,24,(char *) p);
      free(p);
    raton.inicia();
     }
```

```
if((raton.x > 8)&&(raton.x<15)&&(raton.y>15)&&(raton.y<17))//carrera 4
   {
     b1.pon();e1.pon(); e2.pon(); e3.pon(); e5.pon(); e6.pon();
   e4.ilumina(3,15);
   xpos=4;
   for(i=0;i<6;i++)
     a1.escribe(18,7+(i)*3,15,5,opcion[i]);
   a1.escribe(18,7+(xpos-1)*3,11,5,opcion[xpos-1]);
     e4.push(3,15);
     while(raton.boton==1)
       raton.activa();
     raton.desactiva();
   e4.pon();
   e4.ilumina(3,15);
   p = (unsigned char *) malloc((160) * (25)); /*asignar memoria para el buffer de video*/
     salva_pant(0,0,79,24,(char *) p);
   graficaper(valorper);
   ordenaper(valorper,mejorper);
     restaura_pant(0,0,79,24,(char *) p);
     free(p);
   raton.inicia();
   }
if((raton.x > 8)&&(raton.x<15)&&(raton.y>18)&&(raton.y<20))//carrera 5
   {
     b1.pon();e1.pon(); e2.pon(); e3.pon(); e4.pon(); e6.pon();
   e5.ilumina(3,15);
   xpos=5;
   for(i=0;i<6;i++)
     a1.escribe(18,7+(i)*3,15,5,opcion[i]);
   a1.escribe(18,7+(xpos-1)*3,11,5,opcion[xpos-1]);
     e5.push(3,15);
     while(raton.boton==1)
       raton.activa();
     raton.desactiva();
   e5.pon();
   e5.ilumina(3,15);
   p = (unsigned char *) malloc((160) * (25)); /*asignar memoria para el buffer de video*/
     salva_pant(0,0,79,24,(char *) p);
   mainfinal(mejorper);
     restaura_pant(0,0,79,24,(char *) p);
     free(p);
   raton.inicia();
   }
if((raton.x > 8)&&(raton.x<15)&&(raton.y>21)&&(raton.y<23))//lista de carreras
   {
     b1.pon();e1.pon(); e2.pon(); e3.pon(); e4.pon(); e5.pon();
   e6.ilumina(3,15);
   xpos=6;
   for(i=0;i<6;i++)
     a1.escribe(18,7+(i)*3,15,5,opcion[i]);
   a1.escribe(18,22,11,5,opcion[5]);
     e6.push(3,15);
     while(raton.boton==1)
       raton.activa();
     raton.desactiva();
   e6.pon();
```

```
        e6.ilumina(3,15);
        p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
           salva_pant(0,0,79,24,(char *) p);
        imprime(valorper);
        // imprime
           restaura_pant(0,0,79,24,(char *) p);
           free(p);
        raton.inicia();
         }
     }
   } while(continua=='1');
   return(0);
}



// Sistema Experto
int sistema_experto()
{
 char mejorih[4];
 char mejorper[4];
 int valorper[6];
 proceso1(mejorih);
 proceso2(mejorih,valorper,mejorper);
 proceso3();
 proceso4(mejorper);
 menufinal(mejorih, mejorper, valorper);
 return(1);
}
```

ARCHIVO INTRO.CPP.- Es parte de la interface con el usuario.
// Despliega la introduccion inicial al Sistema Experto.

```
#include <c:\tesis\marco.h>

int introduce(char archi[13]); // despliega la introduccion.


int introduce(char archi[13])
{
  register int xpos;
  mouse raton;
  char continua, car;
  pantalla d1(2,1,77,23,15,5,3,1,archi);
  botton e2(35,21,50,22,1,3,0,5,"ADELANTE");
  botton e1(55,21,70,22,1,3,0,5,"P g.Ant.");
  d1.pon_fondo(1,3);
  d1.pon_marco();
  d1.pon_texto(9,7,1,17,15);
  e2.pon();
  e2.ilumina(3,15);
  e1.pon();
  d1.escribe(33,4,14,5,"INTRODUCCION");
  _setcursortype(_NOCURSOR);
  d1.lhor(29,5,48,1,1);
//  d1.lver(47,4,4,1,1);
  continua='1';
  raton.inicia();
  xpos=1;
  do
  {
    raton.activa();
    if(kbhit())
    {
        car=leetecla();
        if(car=='E')  // enter
        {
          if(xpos==1)
          {
            e2.push(3,15);
            delay(100);
            e2.pon();
            raton.desactiva();
            return(1);
          }
          if(xpos==2)
          {
            e1.push(3,15);
            delay(100);
            e1.pon();
            raton.desactiva();
            return(0);
          }
```

```
        }
        if((car=='C')&&(xpos==1))  // derecha
        {
          e2.pon();
          e1.ilumina(3,15);
          xpos=2;
        }
        if((car=='D')&&(xpos==2))  // izquierda
        {
          e1.pon();
          e2.ilumina(3,15);
          xpos=1;
        }
    }
    if(raton.boton==1)
    {
        if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
        {
          e2.push(3,15);
          while(raton.boton==1)
            raton.activa();
          e2.pon();
          e1.pon();
          raton.desactiva();
          return(1);
        }
        if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // cancelar
        {
          e2.pon();
          e1.push(3,15);
          while(raton.boton==1)
            raton.activa();
          e1.pon();
          e2.pon();
          raton.desactiva();
          return(0);
        }
    }
  } while(continua=='1');
  return(0);
}

int main_intro()
{
  while(!introduce("INTRO1.TXT"));
  while(!introduce("INTRO2.TXT"))
    while(!introduce("INTRO1.TXT"));
  while(!introduce("INTRO3.TXT"))
    while(!introduce("INTRO2.TXT"))
        while(!introduce("INTRO1.TXT"));
  while(!introduce("INTRO4.TXT"))
    while(!introduce("INTRO3.TXT"))
        while(!introduce("INTRO2.TXT"))
          while(!introduce("INTRO1.TXT"));
  return(1);
}
```

ARCHIVO DATOS.CPP.- Este archivo contiene funciones que son parte de la interface con el usuario. En particular, se encarga de aplicar el cuestionario de datos personales, lo cual lo hace por medio de la funci¢n datos().

```
// Aplica el cuestionario de datos personales
#include "c:\tesis\marco.h"

int pon(int act, char car);  // Lee caracteres en un fondo negro.


int pon(int x, int y, char car, int pos)
{
  gotoxy(x+1+pos,y+1);
  textcolor(15);
  textbackground(0);
  cprintf("%c",car);
  return(1);
}


int datos()
{
  typedef struct
  {
    int x,y,tam;
  } opciones;
  opciones opcion[9];
  char nombre[41], fecha[9],  edad[3], sexo[11];
  char estado[11], escolaridad[21], ocupa[41];
  char domicilio[58], telefono[11];
  register int xpos, act,i;
  FILE *fp;
  int positem;
  char continua, car;
  mouse raton;
  // limpiar los arreglos
  for(i=0;i<40;i++)
    nombre[i]=' ';
  nombre[40]='\0';
  for(i=0;i<8;i++)
    fecha[i]=' ';
  fecha[8]='\0';
  for(i=0;i<2;i++)
    edad[i]=' ';
  edad[2]='\0';
  for(i=0;i<10;i++)
  {
    sexo[i]=' ';
    estado[i]=' ';
    telefono[i]=' ';
  }
  sexo[10]='\0';
```

```
estado[10]='\0';
telefono[10]='\0';
for(i=0;i<20;i++)
  escolaridad[i]=' ';
escolaridad[20]='\0';
for(i=0;i<40;i++)
  ocupa[i]=' ';
ocupa[40]='\0';
for(i=0;i<57;i++)
  domicilio[i]=' ';
domicilio[57]='\0';
opcion[0].x=15; opcion[0].y=7;  opcion[0].tam=40;
opcion[1].x=28; opcion[1].y=9;  opcion[1].tam=8;
opcion[2].x=13; opcion[2].y=11;  opcion[2].tam=2;
opcion[3].x=34; opcion[3].y=11;  opcion[3].tam=10;
opcion[4].x=62; opcion[4].y=11;  opcion[4].tam=10;
opcion[5].x=20; opcion[5].y=13;  opcion[5].tam=20;
opcion[6].x=18; opcion[6].y=15;  opcion[6].tam=40;
opcion[7].x=18; opcion[7].y=17;  opcion[7].tam=57;
opcion[8].x=17; opcion[8].y=19;  opcion[8].tam=10;
_setcursortype(_NOCURSOR);
pantalla a1(2,1,77,23,15,5,3,1,"DATPER.txt");
botton b2(55,21,70,22,1,3,0,5,"ACEPTAR");
a1.pon_fondo(1,3);
a1.pon_marco();
a1.pon_texto(8,8,1,17,14);
b2.pon();
_setcursortype(_NOCURSOR);
a1.escribe(30,4,14,5,"DATOS PERSONALES");
a1.lhor(29,5,47,1,1);
a1.lver(47,4,4,1,1);
continua='1';
raton.inicia();
positem=0;
act=0;
a1.pon_cursor(opcion[act].x,opcion[act].y,opcion[act].x+opcion[act].tam,opcion[act].y,13,0);
xpos=1;
do
{
  raton.activa();
  if(kbhit())
  {
    car=getch();
    if(kbhit())
      car=getch();
    if(car=='\t')  // tabulador
    {
      positem=0;    // lleva la cuenta de la posicion a leer
        if(act==8)
        {
          b2.pon();
          b2.ilumina(3,15);
          a1.quita_cursor(opcion[act].x,opcion[act].y,opcion[act].x+opcion[act].tam,opcion[act].y);
          act++;
        }
        else
```

```
            if(act==9)
            {
              b2.pon();
              act=0;

a1.pon_cursor(opcion[act].x,opcion[act].y,opcion[act].x+opcion[act].tam,opcion[act].y,13,0);
            }
            else
            {

a1.quita_cursor(opcion[act].x,opcion[act].y,opcion[act].x+opcion[act].tam,opcion[act].y);
              act++;

a1.pon_cursor(opcion[act].x,opcion[act].y,opcion[act].x+opcion[act].tam,opcion[act].y,13,0);
            }
      }
      if(car=='\r')   // enter
      {
        if(act==9)
        {
          b2.push(3,15);
          delay(100);
          b2.pon();
          raton.desactiva();
          continua='0';
        }
      }
      else
        if((isalpha(car))||(isdigit(car))||(ispunct(car))||(car==' ')||(car=='\b'))
        {
          if(car=='\b')    // BACKSPACE
          {
            if(positem!=0)
            {
              positem--;
              car=' ';
              pon(opcion[act].x,opcion[act].y,car,positem);
              switch(act)
              {
                case 0: nombre[positem]=car; nombre[positem+1]='\0'; break;
                case 1: fecha[positem]=car;  fecha[positem+1]='\0'; break;
                case 2: edad[positem]=car;   edad[positem+1]='\0'; break;
                case 3: sexo[positem]=car;   sexo[positem+1]='\0'; break;
                case 4: estado[positem]=car; estado[positem+1]='\0'; break;
                case 5: escolaridad[positem]=car; escolaridad[positem+1]='\0'; break;
                case 6: ocupa[positem]=car;  ocupa[positem+1]='\0'; break;
                case 7: domicilio[positem]=car; domicilio[positem+1]='\0'; break;
                case 8: telefono[positem]=car; telefono[positem+1]='\0'; break;
              }
            }
          }
          else
            if(positem<=(opcion[act].tam -1))
            {
              pon(opcion[act].x,opcion[act].y,car,positem);
              switch(act)
```

```
            {
              case 0: nombre[positem]=car; nombre[positem+1]='\0'; break;
              case 1: fecha[positem]=car;  fecha[positem+1]='\0'; break;
              case 2: edad[positem]=car;   edad[positem+1]='\0'; break;
              case 3: sexo[positem]=car;   sexo[positem+1]='\0'; break;
              case 4: estado[positem]=car; estado[positem+1]='\0'; break;
              case 5: escolaridad[positem]=car; escolaridad[positem+1]='\0'; break;
              case 6: ocupa[positem]=car;  ocupa[positem+1]='\0'; break;
              case 7: domicilio[positem]=car; domicilio[positem+1]='\0'; break;
              case 8: telefono[positem]=car; telefono[positem+1]='\0'; break;
            }
            positem++;
          }
        }
    }
    if(raton.boton==1)
    {
      if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))   // aceptar
      {
        b2.push(3,15);
        while(raton.boton==1)
          raton.activa();
        b2.pon();
        raton.desactiva();
        continua='0';
      }
    }
  } while(continua=='1');
  // ALMACENAR DATOR PERSONALES EN EL ARCHIVO DATPER.DAT
  if((fp=fopen("DATPER.DAT","w"))==NULL)   // crea el archivo para
  {                                        // almacenar los datos
    error(1,"DATPER.DAT");
    return(0);
  }
  fprintf(fp,"%s\n%s\n%s\n%s\n%s\n",nombre, fecha, edad, sexo, estado);
  fprintf(fp,"%s\n%s\n%s\n%s\n",escolaridad, ocupa, domicilio, telefono);
  fclose(fp);
  return(1);
}
```

ARCHIVO CUESTION.CPP.- Las funciones contenidas en este archivo forman parte de la interface con el usuario, especificamente dichas funciones aplican el cuestionario de intereses, recopilando los datos y exhibiendo resultados parciales.

```
// Aplica el cuestionario de intereses
#include "c:\tesis\marco.h"

void cuest1(int interes [60]); // Lleva el control del cuestionario
char lee_interes(pantalla c1, botton b1, botton b2,int interes[60],int pasada);
  // Lee los datos introducidos por el usuario
char cambia_panta(pantalla c1, botton b1, botton b2, int interes[60], int pasada);
  // Se encarga de cambiar la pantalla en el cuestionario
int introducc1();  // Despliega la primera introducci¢n
int introducc2();  // Despliega la segunda introducci¢n
void evalint(int interes [60], int evinter[10]); // Evalua las
     // respuestas recibidas y llena el arreglo de intereses
int graficaint(int evinter[10]); // Grafica el perfil de intereses
int ordenaint(int evinter[10]); // Pone los tres mejores puntos de interes
int almacena(int evinter[10]);
  // almacena los resultados (evinter) en el archivo memoria.dat


char lee_interes(pantalla c1, botton b1, botton b2,int interes[60],int pasada)
// regresa '0' en caso de p gina anterior y '1' en caso de aceptar
// boton b1 es p gina anterior, boton b2 es aceptar
{
  mouse raton;
  int num;
  register int xpos, ypos;
  char *fin, cancela, acepta;
  register int i,interpos, totpos, acpos;
  char car;
  _setcursortype(_NOCURSOR);
  ypos=12;
  xpos=1;
  cancela='0';
  acepta='0';
  c1.pon_cursor(7,ypos,9,ypos,15,0);
  b1.pon();
  b2.pon();
  raton.inicia();
  switch(pasada) {
    case 0: interpos=0; break;
    case 1: interpos=8; break;
    case 2: interpos=16; break;
    case 3: interpos=24; break;
    case 4: interpos=32; break;
    case 5: interpos=40; break;
    case 6: interpos=48; break;
    case 7: interpos=50; break;
    case 8: interpos=58; break;
  }
  totpos=0;
```

```
for(i=12;i<20;i++)
  if((car=que_hay(8,i+1))=='[')  // i+1 indica la posicion actual
  {
    gotoxy(9,i+1);
    printf("%d",interes[interpos+totpos]);
    totpos++;   // totpos es el n&mero de elementos en pantalla
  }
do
{
  raton.inicia();
  do
  {
    if(xpos==1)
      c1.pon_cursor(7,ypos ,9,ypos,15,0);
    raton.activa();
    if(raton.boton==1)
    {
      if((raton.x>55)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))
      {
        b2.pon();
        b1.push(3,15);
        while(raton.boton==1)
          raton.activa();
        b1.pon();
        b2.pon();
        cancela='1';
      }
      if((raton.x>35)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))
      {
        b1.pon();
        b2.push(3,15);
        while(raton.boton==1)
          raton.activa();
        b2.pon();
        b1.pon();
        acepta='1';
      }
      if((pasada==6)||(pasada==8))
      {
        if((raton.x>7)&&(raton.x<11)&&(raton.y>12)&&(raton.y<15))
        {
          c1.quita_cursor(7,ypos,9,ypos);
          ypos=raton.y-1;
          c1.pon_cursor(7,ypos ,9,ypos,15,0);
        }
      }
      else
        if((raton.x>7)&&(raton.x<11)&&(raton.y>12)&&(raton.y<21))
        {
          c1.quita_cursor(7,ypos,9,ypos);
          ypos=raton.y-1;
          c1.pon_cursor(7,ypos ,9,ypos,15,0);
        }
    }
  } while((!kbhit())&&(acepta=='0')&&(cancela=='0'));
      raton.desactiva();
```

```
if(kbhit())
{
  car=leetecla();
  if((car=='0')||(car=='1')||(car=='2')||(car=='3')||(car=='4'))
  {
    if(xpos==1)
    {
acpos=0;
for(i=12;i<=ypos;i++)
{
 if(que_hay(8,i+1)=='[')
 acpos++;
}
num=strtol(&car,&fin,10);     // meter al arreglo de datos
interes[interpos+acpos-1]=num;
gotoxy(9,ypos+1);
putch(car);
    }
  }
  else
    switch(car)
    {
case 'A':{  if(ypos==12)  // arriba
          ypos=12;
          else
          if(xpos==1)
          {
            c1.quita_cursor(7,ypos,9,ypos);
            ypos--;
            c1.pon_cursor(7,ypos,9,ypos,15,0);
          }
        }
        break;
case 'B':{  // abajo
          if(((pasada==6)||(pasada==8))&&(ypos==13))
            ypos=13;
          else
          {
            if(ypos==19)
              ypos=19;
            else
              if(xpos==1)
              {
                c1.quita_cursor(7,ypos,9,ypos);
                ypos++;
                c1.pon_cursor(7,ypos,9,ypos,15,0);
              }
          }
        }
        break;    //derecha
case 'C':   if((ypos==19)||(((pasada==6)||(pasada==8))&&(ypos==13)))
          {
            if(xpos==1)
            {
              xpos++;
              c1.quita_cursor(7,ypos,9,ypos);
```

```
                b2.ilumina(3,15);
              }
              else
               if(xpos==2)
               {
                 xpos++;
                 b2.pon();
                 b1.ilumina(3,15);
               }
            }
            break;
      case 'D':  if((ypos==19)||(((pasada==6)||(pasada==8))&&(ypos==13)))     // izquierda
            {
              if(xpos==3)
              {
                xpos--;
                b1.pon();
                b2.ilumina(3,15);
              }
              else
                if(xpos==2)
                {
                  xpos--;
                  b2.pon();
                  c1.pon_cursor(7,ypos,9,ypos,15,0);
                }
            }
            break;
      case 'E':{  if(xpos==2)//&&(ypos==19)    // enter
            {
              b2.push(3,15);
              delay(200);
              b2.pon();
              b2.ilumina(3,15);
              delay(100);
              b2.pon();
              b1.pon();
              acepta='1';
            }
            else
              if(xpos==3)//&&(ypos==19))
              {
                b1.push(3,15);
                delay(200);
                b1.pon();
                b1.ilumina(3,15);
                delay(100);
                b1.pon();
                b2.pon();
                cancela='1';
              }
              else

if(((ypos==19)&&(xpos==1))||(((pasada==6)||(pasada==8))&&(ypos==13)&&(xpos==1)))
                {
                  c1.quita_cursor(7,ypos,9,ypos);
```

```
                    xpos=2;
                    b2.ilumina(3,15);
                  }
                  else
                    if(xpos==1)
                    {
                      c1.quita_cursor(7,ypos,9,ypos);
                      ypos++;
                      c1.pon_cursor(7,ypos,9,ypos,15,0);
                    }
              }
              break;
            }
        }
  } while((cancela=='0')&&(acepta=='0'));
  if(cancela=='1')
    return('0');
  else
    return('1');
}



char cambia_panta(pantalla c1, botton b1, botton b2, int interes[60], int pasada)
{
  borra(7,12,73,19);
  if(pasada==6)
    c1.pon_texto(8,13,pasada*8 +1,2,15);
  else
    if(pasada==7)
      c1.pon_texto(8,13,51,8,15);
    else
      if(pasada==8)
        c1.pon_texto(8,13,59,2,15);
      else
        c1.pon_texto(8,13,pasada*8 +1,8,15);
  b2.pon();
  cambia(7,11,10,21,7,5);
  return(lee_interes(c1,b1,b2,interes, pasada));
}



void cuest1(int interes [60])
{
  register int i;
  pantalla c1(2,1,77,23,15,5,3,1,"interes.txt");
  botton b2(35,21,50,22,1,3,0,5,"ADELANTE");
  botton b1(55,21,70,22,1,3,0,5,"P g.Ant.");
  clrscr();
  c1.pon_modo(3);
  c1.pon_fondo(1,3);
  c1.pon_grueso();
  _setcursortype(_NOCURSOR);
  c1.escribe(28,4,14,5,"CUESTIONARIO DE INTERESES");
```

```
  c1.lhor(27,5,53,1,1);
  c1.lhor(44,6,68,1,0);
  c1.lhor(44,12,68,1,0);
  c1.escribe(45,7,15,5,"4 = me gustaría mucho");
  c1.escribe(45,8,15,5,"3 = me gustaría un poco");
  c1.escribe(45,9,15,5,"2 = me es indiferente");
  c1.escribe(45,10,15,5,"1 = me desagrada un poco");
  c1.escribe(45,11,15,5,"0 = me desagrada mucho");
  cambia(7,11,10,21,7,5);
  b1.pon();
  b2.pon();
  for(i=0;i<60;i++)     // limpia el arreglo de inter,s
    interes[i]=0;
  for(i=0;i<9;i++)
  {
        if(i>6)
        {
          c1.escribe(8,7,14,5,"Ahora piensa en esta pregunta: ");
          c1.escribe(8,8,14,5,"                              ");
          c1.escribe(8,9,14,5,"¨Qu, tanto me gustaría     ");
          c1.escribe(8,10,14,5,"trabajar como ...?        ");
        }
        else
        {
          c1.escribe(8,7,14,5,"Al ir respondiendo piensa en la");
          c1.escribe(8,8,14,5,"siguiente pregunta:        ");
          c1.escribe(8,9,14,5,"                           ");
          c1.escribe(8,10,14,5,"¨Qu, tanto me gustaría...?  ");
        }
        if(cambia_panta(c1,b1,b2,interes,i)=='1');  // cambia pantalla
        else
          if(i==0)
            i=-1;
          else
            if(i>-1)
                i-=2;
  }

}


int introducc1()
{
  register int xpos;
  char continua, car;
  mouse raton;
  _setcursortype(_NOCURSOR);
  pantalla a1(2,1,77,23,15,5,3,1,"int3.txt");
  botton b2(35,21,50,22,1,3,0,5,"ADELANTE");
  botton b1(55,21,70,22,1,3,0,5,"P g.Ant.");
  a1.pon_fondo(1,3);
  a1.pon_marco();
  a1.pon_texto(8,6,1,17,15);
  b2.pon();
  b2.ilumina(3,15);
  b1.pon();
```

```
_setcursortype(_NOCURSOR);
a1.escribe(30,4,14,5,"MODULO INTERESES");
a1.lhor(29,5,47,1,1);
a1.lver(47,4,4,1,1);
continua='1';
raton.inicia();
xpos=1;
do
{
  raton.activa();
  if(kbhit())
  {
    car=leetecla();
    if(car=='E')  // enter
    {
      if(xpos==1)
      {
        b2.push(3,15);
        delay(100);
        b2.pon();
        raton.desactiva();
        return(1);
      }
      if(xpos==2)
      {
        b1.push(3,15);
        delay(100);
        b1.pon();
        raton.desactiva();
        return(0);
      }
    }
    if((car=='C')&&(xpos==1))  // derecha
    {
      b2.pon();
      b1.ilumina(3,15);
      xpos=2;
    }
    if((car=='D')&&(xpos==2))  // izquierda
    {
      b1.pon();
      b2.ilumina(3,15);
      xpos=1;
    }
  }
  if(raton.boton==1)
  {
    if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
    {
      b2.push(3,15);
      while(raton.boton==1)
        raton.activa();
      b2.pon();
      raton.desactiva();
      return(1);
    }
```

```
      if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // cancelar
      {
        b2.pon();
        b1.push(3,15);
        while(raton.boton==1)
        raton.activa();
        b1.pon();
        raton.desactiva();
        return(0);
      }
    }
  } while(continua=='1');
  return(0);
}



int introducc2()
{
  register int xpos;
  mouse raton;
  char continua, car;
  pantalla d1(2,1,77,23,15,5,3,1,"int31.txt");
  botton e2(35,21,50,22,1,3,0,5,"ADELANTE");
  botton e1(55,21,70,22,1,3,0,5,"P g.Ant.");
  d1.pon_fondo(1,3);
  d1.pon_marco();
  d1.pon_texto(11,8,1,17,15);
  e2.pon();
  e2.ilumina(3,15);
  e1.pon();
  d1.escribe(30,4,14,5,"MODULO INTERESES");
  _setcursortype(_NOCURSOR);
  d1.lhor(29,5,47,1,1);
  d1.lver(47,4,4,1,1);
  continua='1';
  raton.inicia();
  xpos=1;
  do
  {
    raton.activa();
    if(kbhit())
    {
      car=leetecla();
      if(car=='E')  // enter
      {
        if(xpos==1)
        {
          e2.push(3,15);
          delay(100);
          e2.pon();
          raton.desactiva();
          return(1);
        }
        if(xpos==2)
```

```
        {
          e1.push(3,15);
          delay(100);
          e1.pon();
          raton.desactiva();
          return(0);
        }
      }
      if((car=='C')&&(xpos==1))  // derecha
      {
        e2.pon();
        e1.ilumina(3,15);
        xpos=2;
      }
      if((car=='D')&&(xpos==2))  // izquierda
      {
        e1.pon();
        e2.ilumina(3,15);
        xpos=1;
      }
    }
    if(raton.boton==1)
    {
      if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
      {
        e2.push(3,15);
        while(raton.boton==1)
          raton.activa();
        e2.pon();
        e1.pon();
        raton.desactiva();
        return(1);
      }
      if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // cancelar
      {
        e2.pon();
        e1.push(3,15);
        while(raton.boton==1)
          raton.activa();
        e1.pon();
        e2.pon();
        raton.desactiva();
        return(0);
      }
    }
  } while(continua=='1');
  return(0);
}


void evalint(int interes [60], int evinter[10])
{
  register int i,j;
  // limpia evinter
  for(i=0;i<10;i++)
```

```
    evinter[i]=0;
  // calcula los resultados
  for(j=0;j<10;j++)
    for(i=0;i<60;i+=10)
        evinter[j]+=interes[i+j];
}

int graficaint(int evinter[10])
{
  unsigned char *p;
  pantemerge a1(15,4,65,21,15,7,0,8,"ayuint.txt");
  mouse raton;
  char continua, car;
  div_t x;
  register int i, j, por, cuadros, ypos;
  pantalla g1(0,0,80,25,5,5,5,5,"          ");
  botton e2(61,18,76,19,1,3,0,5,"ACEPTAR");
  botton e1(61,21,76,22,1,3,0,5," AYUDA");
  g1.pon_marco();
  e2.pon();
  e2.ilumina(3,15);
  e1.pon();
  g1.escribe(32,1,14,5,"PERFIL DE INTERESES");
  g1.escribe(2,4,15,5, "ASISTENCIAL");
  g1.escribe(2,6,15,5, "EJECUTIVO-PERSUASIVO");
  g1.escribe(2,8,15,5, "VERBAL");
  g1.escribe(2,10,15,5, "ARTISTICO-PLASTICO");
  g1.escribe(2,12,15,5,"MUSICAL");
  g1.escribe(2,14,15,5,"ORGANIZACION");
  g1.escribe(2,16,15,5,"CIENTIFICO");
  g1.escribe(2,18,15,5,"CALCULO");
  g1.escribe(2,20,15,5,"MECANICO-CONSTRUCT.");
  g1.escribe(2,22,15,5,"ACT. AL AIRE LIBRE");
  g1.escribe(22,25,15,5,"  10 20 30 40 50 60 70 80 90 100     %");
  g1.escribe(61,4,14,5,"Esta  grafica  es");
  g1.escribe(61,5,14,5,"resultado de  tus");
  g1.escribe(61,6,14,5,"respuestas  en  el");
  g1.escribe(61,7,14,5,"cuestionario   de");
  g1.escribe(61,8,14,5,"intereses.");
  g1.lhor(23,24,61,15,0);
  g1.lver(23,3,23,15,0);
  _setcursortype(_NOCURSOR);
  gotoxy(23,24);  putch('À');
  for(i=0;i<10;i++)
  {
    x=div(evinter[i]*100,24);
    por=x.quot;
    x=div(por*30,100);
    cuadros=x.quot;
    for(j=0;j<cuadros;j++)
    {
      g1.escribe(24+j,4+i*2,9,5,"Û");
      if(j!=0)
        g1.escribe(24+j,4+i*2+1,0,5,"ß");
      if(j==cuadros-1)
```

```
    {
      g1.escribe(24+j+1,4+i*2,0,5,"Ú");
      g1.escribe(24+j+1,4+i*2+1,0,5,"ß");
    }
  }
}
continua='1';
raton.inicia();
ypos=1;
do
{
  raton.activa();
  if(kbhit())
  {
    car=leetecla();
    if(car=='E')  // enter
    {
      if(ypos==1)     // aceptar
      {
        e1.pon();
        e2.push(3,15);
        delay(100);
        e2.pon();
        raton.desactiva();
        return(1);
      }
      if(ypos==2)  // ayuda
      {
        e1.push(3,15);
        delay(100);
        e1.pon();
        raton.desactiva();
        p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
        salva_pant(0,0,79,24,(char *) p);
        a1.pon_marco();
        a1.control(0);
        restaura_pant(0,0,79,24,(char *) p);
        free(p);
        e2.pon();
        e1.ilumina(3,15);
        raton.inicia();
      }
    }
    if((car=='A')&&(ypos==2))  // arriba
    {
      e1.pon();
      e2.ilumina(3,15);
      ypos=1;
    }
    if((car=='B')&&(ypos==1))  // abajo
    {
      e2.pon();
      e1.ilumina(3,15);
      ypos=2;
    }
  }
```

```
    if(raton.boton==1)
    {
      if((raton.x > 60)&&(raton.x<77)&&(raton.y>18)&&(raton.y<20))  // aceptar
      {
        e1.pon();
        e2.push(3,15);
        while(raton.boton==1)
          raton.activa();
        e2.pon();
        e1.pon();
        raton.desactiva();
        return(1);
      }
      if((raton.x > 60)&&(raton.x<77)&&(raton.y>21)&&(raton.y<23))  // ayuda
      {
        e2.pon();
        e1.push(3,15);
        while(raton.boton==1)
        raton.activa();
        e1.pon();
        e2.pon();
        raton.desactiva();
        p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
        salva_pant(0,0,79,24,(char *) p);
        a1.pon_marco();
        a1.control(0);
        restaura_pant(0,0,79,24,(char *) p);
        free(p);
        e1.ilumina(3,15);
        raton.inicia();
        ypos=2;
      }
    }
  } while(continua=='1');
  return(0);
}


int ordenaint(int evinter[10])
// ordena el arreglo de intereses y muestra los tres mejores
{
  unsigned char *p;
  pantemerge a1(15,4,65,21,15,7,0,8,"ayuint.txt");
  mouse raton;
  char continua, car;
  pantalla d1(2,1,77,23,15,5,3,1,"ordenint.txt");
  botton e2(35,21,50,22,1,3,0,5,"ACEPTAR");
  botton e1(55,21,70,22,1,3,0,5,"AYUDA");
  d1.pon_fondo(1,3);
  d1.pon_marco();
  d1.pon_texto(13,8,1,17,15);
  e2.pon();
  e1.pon();
  e2.ilumina(3,15);
  d1.escribe(30,4,14,5,"PERFIL DE INTERESES");
  _setcursortype(_NOCURSOR);
```

```
d1.lhor(29,5,50,1,1);
d1.lver(50,4,4,1,1);
register int i,j, pasa, xpos;
typedef struct
{
  char elemento[23];
  int puntos;
} arreglo;
arreglo c;
arreglo intereses[10];
// vacia evinter en el nuevo arreglo
for(i=0;i<10;i++)
  intereses[i].puntos=evinter[i];
// vaciar la clave de intereses
strcpy(intereses[0].elemento,"ASISTENCIAL");
strcpy(intereses[1].elemento,"EJECUTIVO-PERSUASIVO");
strcpy(intereses[2].elemento,"VERBAL");
strcpy(intereses[3].elemento,"ARTISTICO-PLASTICO");
strcpy(intereses[4].elemento,"MUSICAL");
strcpy(intereses[5].elemento,"ORGANIZACION");
strcpy(intereses[6].elemento,"CIENTIFICO");
strcpy(intereses[7].elemento,"CALCULO");
strcpy(intereses[8].elemento,"MECANICO-CONSTRUCTIVO");
strcpy(intereses[9].elemento,"ACTIVIDAD AL AIRE LIBRE");
// ordenar el arreglo de intereses, metodo de la burbuja
for(i=0;i<10;i++)
  for(j=9;j>i;j--)
    if (intereses[j-1].puntos < intereses[j].puntos)
    {
      c=intereses[j-1];
      intereses[j-1]=intereses[j];
      intereses[j]=c;
    }
d1.escribe(25,14,14,5,intereses[0].elemento);
d1.escribe(25,16,14,5,intereses[1].elemento);
d1.escribe(25,18,14,5,intereses[2].elemento);
continua='1';
raton.inicia();
xpos=1;
do
{
  raton.activa();
  if(kbhit())
  {
    car=leetecla();
    if(car=='E')  // enter
    {
      if(xpos==1)     // aceptar
      {
        e1.pon();
        e2.push(3,15);
        delay(100);
        e2.pon();
        raton.desactiva();
        return(1);
      }
```

```
    if(xpos==2)      // ayuda
    {
      e1.push(3,15);
      delay(100);
      e1.pon();
      raton.desactiva();
      p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
      salva_pant(0,0,79,24,(char *) p);
      a1.pon_marco();
      a1.control(0);
      restaura_pant(0,0,79,24,(char *) p);
      free(p);
      e2.pon();
      e1.ilumina(3,15);
      raton.inicia();
    }
  }
  if((car=='C')&&(xpos==1))  // derecha
  {
    e2.pon();
    e1.ilumina(3,15);
    xpos=2;
  }
  if((car=='D')&&(xpos==2))  // izquierda
  {
    e1.pon();
    e2.ilumina(3,15);
    xpos=1;
  }
 }
 if(raton.boton==1)
 {
  if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
  {
    e1.pon();
    e2.ilumina(3,15);
    e2.push(3,15);
    while(raton.boton==1)
      raton.activa();
    e2.pon();
    raton.desactiva();
    return(1);
  }
  if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // ayuda
  {
    e2.pon();
    e1.push(3,15);
    while(raton.boton==1)
      raton.activa();
    e1.pon();
    e2.pon();
    raton.desactiva();
    p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
    salva_pant(0,0,79,24,(char *) p);
    a1.pon_marco();
    a1.control(0);
```

```
        restaura_pant(0,0,79,24,(char *) p);
        free(p);
        e1.ilumina(3,15);
        raton.inicia();
        xpos=2;
      }
    }
  } while(continua=='1');
  return(1);
}


int almacena(int evinter[10])
{
   register int i;
   FILE *fp;
   if((fp=fopen("memoria.dat","w"))==NULL)  // crea el archivo para
   {                                       // almacenar los datos
    error(1,"MEMORIA.DAT");
    return(0);
   }
   for(i=0;i<10;i++)
    fprintf(fp,"%d ",evinter[i]);
   fprintf(fp,"\n");
   fclose(fp);
   return(1);
}


int main_interes()
{
  // evinter contiene las diez areas de interes y su puntuacion
  int interes [60];  // contiene las respuestas a las preguntas del cuestionario
  int evinter[10];
  while(!(introducc1()));
   while(!(introducc2()))
     while(!(introducc1()));
  cuest1(interes);
  evalint(interes, evinter);
  graficaint(evinter);
  ordenaint(evinter);
  almacena(evinter);
  return(1);
}


int resulta_int()
{
  FILE *fp;
  int evinter[10];
  register int i;
  if((fp=fopen("memoria.dat","r"))==NULL)  // leer  el archivo de datos
  {
    error(1,"MEMORIA.DAT");
```

```
    return(0);
  }
  for(i=0;i<10;i++)
    fscanf(fp,"%d ",&evinter[i]);
  fscanf(fp,"\n");
  fclose(fp);
  graficaint(evinter);
  ordenaint(evinter);
  return(1);
}
```

```
// ARCHIVO CUES1.CPP.- Las funciones contenidas en este archivo son
// parte de la interface con el usuario.  Especificamente  se encargan
// de aplicar el cuestionario de habilidades.


// Aplica el cuestionario de habilidades
#include "c:\tesis\marco.h"

void cuest1h(int habilidad[60]);  // Lleva el control del cuestionario
char lee_habilidad(pantalla c1, botton b1, botton b2,int habilidad[60],int pasada);
  // Lee los datos introducidos por el usuario
char cambia_pantah(pantalla c1, botton b1, botton b2, int habilidad[60], int pasada);
  // Cambia la pantalla al aplicar el cuestionario
int introducc1h();  // Despliega la primera introduccion
int introducc2h();  // Despliega la segunda introduccion
int ordenahabi(int evhabi[10]);   // Ordena el arreglo de resultados
int graficahabi(int evhabi[10]);  // Grafica el perfil de habilidades
int almacenahabi(int evhabi[10]);
// almacena los datos en memoria de trabajo



char lee_habilidad(pantalla c1, botton b1, botton b2,int habilidad[60],int pasada)
// regresa '0' en caso de p gina anterior y '1' en caso de aceptar
// boton b1 es p gina anterior, boton b2 es aceptar
{
  mouse raton;
  register int xpos, ypos, totpos, acpos;
  int habipos;
  int num;
  char *fin, cancela, acepta;
  register int i;
  char car;
  ypos=13;        // posicion del cursor en pantalla, enpieza en 13
  xpos=1;
  cancela='0';
  acepta='0';
  switch(pasada) {
    case 0: habipos=0; break;
    case 1: habipos=7; break;
    case 2: habipos=12; break;
    case 3: habipos=19; break;
    case 4: habipos=24; break;
    case 5: habipos=29; break;
    case 6: habipos=35; break;
    case 7: habipos=40; break;
    case 8: habipos=45; break;
    case 9: habipos=49; break;
    case 10:habipos=54; break;
    case 11:habipos=59; break;
  }
  totpos=0;
  for(i=13;i<20;i++)
    if(que_hay(8,i+1)=='[')  // i+1 indica la posicion actual
    {
```

```
        gotoxy(9,i+1);
        printf("%d",habilidad[habipos+totpos]);
        totpos++;    // totpos es el n£mero de elementos en pantalla
    }
  c1.pon_cursor(7,ypos,9,ypos,15,0);
  raton.inicia();
//  raton.activa();
  do
  { raton.inicia();
    do
    {
        if(xpos==1)
          c1.pon_cursor(7,ypos,9,ypos,15,0);
        raton.activa();
        if(raton.boton==1)
        {
          if((raton.x>55)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))
          // se activ¢ boton p gina anterior
          {
            b2.pon();
            b1.push(3,15);
            while(raton.boton==1)
                raton.activa();
            b1.pon();
            b2.pon();
            cancela='1';
          }
          if((raton.x>35)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))
          // se activ¢ boton acepta
          {
            b1.pon();
            b2.push(3,15);
            while(raton.boton==1)
                raton.activa();
            b2.pon();
            b1.pon();
            acepta='1';
          }
          if((raton.x>7)&&(raton.x<11)&&(raton.y>12)&&(raton.y<21))
            {
                if(((que_hay(raton.x,raton.y) == '[')||
                    (que_hay(raton.x,raton.y)==']'))||
                    ((que_hay(raton.x-1,raton.y)=='[')&&(que_hay(raton.x+1,raton.y)==']')))
                {
                  c1.quita_cursor(7,ypos,9,ypos);
                  ypos=raton.y - 1;
                  c1.pon_cursor(7,ypos,9,ypos,15,0);
                }
            }
        }
    } while((!kbhit())&&(acepta=='0')&&(cancela=='0'));
        raton.desactiva();
        if(kbhit())
        {
          car=leetecla();
          if((car=='0')||(car=='1')||(car=='2')||(car=='3')||(car=='4'))
```

```
{
  if(xpos==1)
  {
       textbackground(0);
       gotoxy(9,ypos+1);
       // contar la posicion de la pregunta actual
       acpos=0;
       for(i=13;i<=ypos;i++)
       {
         if(que_hay(8,i+1)=='[')
              acpos++;
       }
       num=strtol(&car,&fin,10);     // meter al arreglo de datos
       habilidad[habipos+acpos-1]=num;
       putch(car);
  }
}
else
  switch(car)
  {
       case 'A':{  if(ypos==13)  // arriba
                        ypos=13;
                   else
                   if(xpos==1)
                   {
                        if(que_hay(8,ypos)=='[')
                        {
                          c1.quita_cursor(7,ypos,9,ypos);
                          ypos--;
                          c1.pon_cursor(7,ypos ,9,ypos,15,0);
                        }
                        else
                        {
                          if(que_hay(8,ypos-1)=='[')
                          {
                            c1.quita_cursor(7,ypos,9,ypos);
                            ypos=ypos-2;
                            c1.pon_cursor(7,ypos,9,ypos,15,0);
                          }
                        }
                   }
                }
                break;
       case 'B':{  // abajo
                   if(ypos==19)
                     ypos=19;
                   else
                   if(xpos==1)
                   {
                     if(que_hay(8,ypos+2)=='[')
                     {
                       c1.quita_cursor(7,ypos,9,ypos);
                       ypos++;
                       c1.pon_cursor(7,ypos,9,ypos,15,0);
                     }
                     else
```

```
                    {
                      if(que_hay(8,ypos+3)=='[')
                      {
                          c1.quita_cursor(7,ypos,9,ypos);
                          ypos+=2;
                          c1.pon_cursor(7,ypos,9,ypos,15,0);
                      }
                    }
                  }
              }
              break;    //derecha
case 'C': {
              if(ypos==19)
              {
                  if(xpos==1)
                  {
                    xpos++;
                    c1.quita_cursor(7,ypos,9,ypos);
                    b2.ilumina(3,15);
                  }
                  else
                    if(xpos==2)
                    {
                      xpos++;
                      b2.pon();
                      b1.ilumina(3,15);
                    }
              }
              else
              {
                  if((que_hay(8,ypos+2)!='[')&&(que_hay(8,ypos+3)!='['))
                  {
                    xpos++;
                    c1.quita_cursor(7,ypos,9,ypos);
                    b2.ilumina(3,15);
                    ypos=19;
                  }
              }
            }
            break;
case 'D':   if(ypos==19)     // Izquierda
            {
                  if(xpos==3)
                  {
                    xpos--;
                    b1.pon();
                    b2.ilumina(3,15);
                  }
                  else
                    if(xpos==2)
                    {
                      xpos--;
                      b2.pon();
                      while(que_hay(8,ypos+1)!='[')
                            ypos--;
                      c1.pon_cursor(7,ypos ,9,ypos,15,0);
```

```
                    }
                }
                break;
        case 'E':{  if(xpos==2)//&&(ypos==8))    // enter
                    {
                        b2.push(3,15);
                        delay(200);
                        b2.pon();
                        b2.ilumina(3,15);
                        delay(100);
                        b2.pon();
                        acepta='1';
                    }
                    else
                        if(xpos==3)//&&(ypos==8))
                        {
                          b1.push(3,15);
                          delay(200);
                          b1.pon();
                          b1.ilumina(3,15);
                          delay(100);
                          b1.pon();
                          cancela='1';
                        }
                          else
                          {
                            if(ypos==19)
                            {
                                xpos++;
                                c1.quita_cursor(7,ypos,9,ypos);
                                b2.ilumina(3,15);
                                ypos=19;
                            }
                            else
                                if(xpos==1)
                                {
                                  if(que_hay(8,ypos+2)=='[')
                                  {
                                      c1.quita_cursor(7,ypos,9,ypos);
                                      ypos++;
                                      c1.pon_cursor(7,ypos,9,ypos,15,0);
                                  }
                                  else
                                  {
                                       if(que_hay(8,ypos+3)=='[')
                                       {
                                         c1.quita_cursor(7,ypos,9,ypos);
                                         ypos+=2;
                                         c1.pon_cursor(7,ypos,9,ypos,15,0);
                                       }
                                       else
                                       {
if((que_hay(8,ypos+2)!='[')&&(que_hay(8,ypos+3)!='['))
                                           {
                                             xpos++;
```

```
                                                    c1.quita_cursor(7,ypos,9,ypos);
                                                    b2.ilumina(3,15);
                                                    ypos=19;
                                                  }
                                                }
                                              }
                                            }
                                          }
                                        }
                                        break;
                                    }
                                }
} while((cancela=='0')&&(acepta=='0'));
//  raton.desactiva();
  if(cancela=='1')
    return('0');
  else
    return('1');
}



char cambia_pantah(pantalla c1, botton b1, botton b2, int habilidad[60], int pasada)
{
  // desaparecer el cursor
  borra(7,12,73,19);
  if(pasada==11)
    c1.pon_texto(8,14,pasada*7 +1,2,15);
  else
    c1.pon_texto(8,14,pasada*7 +1,7,15);
  b2.pon();
  cambia(7,12,10,21,7,5);
  return(lee_habilidad(c1,b1,b2,habilidad, pasada));
}



void cuest1h(int habilidad[60])
{
  register int i;
  pantalla c1(2,1,77,23,15,5,3,1,"HABIL.TXT");
  botton b2(35,21,50,22,1,3,0,5,"ADELANTE");
  botton b1(55,21,70,22,1,3,0,5,"P g.Ant.");
  clrscr();
  c1.pon_modo(3);
  c1.pon_fondo(1,3);
  c1.pon_grueso();
  c1.escribe(27,4,14,5,"CUESTIONARIO DE HABILIDADES");
  c1.lhor(26,5,54,1,1);
  c1.lhor(52,6,75,1,0);
  c1.lhor(52,12,75,1,0);
  c1.escribe(53,7,15,5,"4 = muy h bil");
  c1.escribe(53,8,15,5,"3 = h bil");
  c1.escribe(53,9,15,5,"2 = medianamente h bil");
  c1.escribe(53,10,15,5,"1 = poco h bil");
```

```
  c1.escribe(53,11,15,5,"0 = nada h bil");
  c1.escribe(8,7,14,5,"Al ir respondiendo piensa en la");
  c1.escribe(8,8,14,5,"siguiente pregunta:    ");
  c1.escribe(8,10,14,5,"Qu, tan h bil me considero para....?");
  c1.escribe(8,11,14,5,"(independientemente de que me guste o no)");
  _setcursortype(_NOCURSOR);
  cambia(7,12,10,21,7,5);
  b1.pon();
  b2.pon();
  for(i=0;i<60;i++)     // limpia el arreglo de inter,s
    habilidad[i]=0;
  for(i=0;i<12;i++)
  {
        if(cambia_pantah(c1,b1,b2,habilidad,i)=='1'); // cambia pantalla
        else
          if(i==0)
            i=-1;
          else
            if(i>-1)
                i-=2;
  }
}


int introducc1h()
{
  int xpos;
  char car;
  char continua;
  mouse raton;
  _setcursortype(_NOCURSOR);
  pantalla a1(2,1,77,23,15,5,3,1,"int3h.txt");
  botton b2(35,21,50,22,1,3,0,5,"ADELANTE");
  botton b1(55,21,70,22,1,3,0,5,"P g.Ant.");
  a1.pon_fondo(1,3);
  a1.pon_marco();
  a1.pon_texto(8,5,1,17,15);
  b2.pon();
  b2.ilumina(3,15);
  b1.pon();
  a1.escribe(29,3,14,5,"MODULO HABILIDADES");
  a1.lhor(28,4,48,1,1);
  a1.lver(48,3,3,1,1);
  continua='1';
  raton.inicia();
  xpos=1;
  do
  {
    raton.activa();
    if(kbhit())
    {
        car=leetecla();
        if(car=='E')  // enter
        {
          if(xpos==1)
          {
```

```
            b2.push(3,15);
            delay(100);
            b2.pon();
            raton.desactiva();
            return(1);
          }
          if(xpos==2)
          {
            b1.push(3,15);
            delay(100);
            b1.pon();
            raton.desactiva();
            return(0);
          }
        }
        if((car=='C')&&(xpos==1))  // derecha
        {
          b2.pon();
          b1.ilumina(3,15);
          xpos=2;
        }
        if((car=='D')&&(xpos==2))  // izquierda
        {
          b1.pon();
          b2.ilumina(3,15);
          xpos=1;
        }
    }
    if(raton.boton==1)
    {
        if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
        {
          b2.push(3,15);
          while(raton.boton==1)
            raton.activa();
          b2.pon();
          raton.desactiva();
          return(1);
        }
        if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // cancelar
        {
          b2.pon();
          b1.push(3,15);
          while(raton.boton==1)
            raton.activa();
          b1.pon();
          raton.desactiva();
          return(0);
        }
    }
  } while(continua=='1');
  return(0);
}

int introducc2h()
```

```
{
  int xpos;
  char car;
  mouse raton;
  char continua;
  _setcursortype(_NOCURSOR);
  pantalla d1(2,1,77,23,15,5,3,1,"int31h.txt");
  botton e2(35,21,50,22,1,3,0,5,"ADELANTE");
  botton e1(55,21,70,22,1,3,0,5,"P g.Ant.");
  d1.pon_fondo(1,3);
  d1.pon_marco();
  d1.pon_texto(11,8,1,17,15);
  e2.pon();
  e2.ilumina(3,15);
  e1.pon();
  d1.escribe(29,4,14,5,"MODULO HABILIDADES");
  d1.lhor(28,5,48,1,1);
  d1.lver(48,4,4,1,1);
  continua='1';
  raton.inicia();
  xpos=1;
  do
  {
    raton.activa();
    if(kbhit())
    {
        car=leetecla();
        if(car=='E')  // enter
        {
          if(xpos==1)
          {
            e2.push(3,15);
            delay(100);
            e2.pon();
            raton.desactiva();
            return(1);
          }
          if(xpos==2)
          {
            e1.push(3,15);
            delay(100);
            e1.pon();
            raton.desactiva();
            return(0);
          }
        }
        if((car=='C')&&(xpos==1))  // derecha
        {
          e2.pon();
          e1.ilumina(3,15);
          xpos=2;
        }
        if((car=='D')&&(xpos==2))  // izquierda
        {
          e1.pon();
          e2.ilumina(3,15);
```

```
            xpos=1;
          }
    }
    if(raton.boton==1)
    {
          if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
          {
            e2.push(3,15);
            while(raton.boton==1)
              raton.activa();
            e2.pon();
            raton.desactiva();
            return(1);
          }
          if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // cancelar
          {
            e2.pon();
            e1.push(3,15);
            while(raton.boton==1)
              raton.activa();
            e1.pon();
            raton.desactiva();
            return(0);
          }
    }
  } while(continua=='1');
  return(0);
}


void evalhabi(int habilidad [60], int evhabi[10])
{
  register int i,j;
  // limpia evhabi
  for(i=0;i<10;i++)
    evhabi[i]=0;
  // calcula los resultados
  for(j=0;j<10;j++)
    for(i=0;i<60;i+=10)
          evhabi[j]+=habilidad[i+j];
}


int graficahabi(int evhabi[10])
{
  unsigned char *p;
  pantemerge a1(15,4,65,21,15,7,0,8,"ayuhabi.txt");
  mouse raton;
  char continua, car;
  div_t x;
  register int i, j, por, cuadros, ypos;
  pantalla g1(0,0,80,25,5,5,5,5,"          ");
  botton e2(61,18,76,19,1,3,0,5,"ACEPTAR");
  botton e1(61,21,76,22,1,3,0,5," AYUDA");
  g1.pon_marco();
```

```
e2.pon();
e2.ilumina(3,15);
e1.pon();
g1.escribe(31,1,14,5,"PERFIL DE HABILIDADES");
g1.escribe(2,4,15,5, "SERVICIO SOCIAL");
g1.escribe(2,6,15,5, "EJECUTIVO-PERSUASIVO");
g1.escribe(2,8,15,5, "VERBAL");
g1.escribe(2,10,15,5, "ARTISTICO-PLASTICA");
g1.escribe(2,12,15,5,"MUSICAL");
g1.escribe(2,14,15,5,"ORGANIZACION");
g1.escribe(2,16,15,5,"CIENTIFICA");
g1.escribe(2,18,15,5,"CALCULO");
g1.escribe(2,20,15,5,"MECANICA-CONSTRUCT.");
g1.escribe(2,22,15,5,"DESTREZA MANUAL");
g1.escribe(22,25,15,5," 10 20 30 40 50 60 70 80 90 100     %");
g1.escribe(61,4,14,5,"Esta  grafica  es");
g1.escribe(61,5,14,5,"resultado de tus");
g1.escribe(61,6,14,5,"respuestas en el");
g1.escribe(61,7,14,5,"cuestionario  de");
g1.escribe(61,8,14,5,"habilidades.");
g1.lhor(23,24,61,15,0);
g1.lver(23,3,23,15,0);
_setcursortype(_NOCURSOR);
gotoxy(23,24); putch('Á');
for(i=0;i<10;i++)
{
  x=div(evhabi[i]*100,24);
  por=x.quot;
  x=div(por*30,100);
  cuadros=x.quot;
  for(j=0;j<cuadros;j++)
  {
       g1.escribe(24+j,4+i*2,9,5,"Ü");
       if(j!=0)
         g1.escribe(24+j,4+i*2+1,0,5,"ß");
       if(j==cuadros-1)
       {
         g1.escribe(24+j+1,4+i*2,0,5,"Ü");
         g1.escribe(24+j+1,4+i*2+1,0,5,"ß");
       }
  }
}
continua='1';
raton.inicia();
ypos=1;
do
{
  raton.activa();
  if(kbhit())
  {
       car=leetecla();
       if(car=='E')  // enter
       {
         if(ypos==1)              // aceptar
         {
           e2.push(3,15);
```

```
            delay(100);
            e2.pon();
            raton.desactiva();
            return(1);
          }
          if(ypos==2)              // ayuda
          {
            e1.push(3,15);
            delay(100);
            e1.pon();
        raton.desactiva();
        p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
            salva_pant(0,0,79,24,(char *) p);
        a1.pon_marco();
        a1.control(0);
        restaura_pant(0,0,79,24,(char *) p);
        free(p);
        e2.pon();
        e1.ilumina(3,15);
        raton.inicia();
          }
        }
        if((car=='A')&&(ypos==2))  // arriba
        {
          e1.pon();
          e2.ilumina(3,15);
          ypos=1;
        }
        if((car=='B')&&(ypos==1))  // abajo
        {
          e2.pon();
          e1.ilumina(3,15);
          ypos=2;
        }
    }
    if(raton.boton==1)
    {
        if((raton.x > 60)&&(raton.x<77)&&(raton.y>18)&&(raton.y<20))  // aceptar
        {
          e1.pon();
       e2.ilumina(3,15);
          e2.push(3,15);
          while(raton.boton==1)
            raton.activa();
          e2.pon();
          e1.pon();
          raton.desactiva();
          return(1);
        }
        if((raton.x > 60)&&(raton.x<77)&&(raton.y>21)&&(raton.y<23))  // ayuda
        {
          e2.pon();
          e1.push(3,15);
          while(raton.boton==1)
            raton.activa();
          e1.pon();
```

```
            e2.pon();
            raton.desactiva();
          p = (unsigned char *) malloc((160) * (24)); /*asignar memoria para el buffer de video*/
            salva_pant(0,0,79,24,(char *) p);
          a1.pon_marco();
          a1.control(0);
          restaura_pant(0,0,79,24,(char *) p);
          free(p);
          e1.ilumina(3,15);
          raton.inicia();
          ypos=2;
           }
      }
    } while(continua=='1');
    return(0);
}



int ordenahabi(int evhabi[10])
// ordena el arreglo de habilidades y muestra los tres mejores
{
    unsigned char *p;
    pantemerge a1(15,4,65,21,15,7,0,8,"ayuhabi.txt");
    mouse raton;
    char continua, car;
    pantalla d1(2,1,77,23,15,5,3,1,"ordenhab.txt");
    botton e2(35,21,50,22,1,3,0,5,"ACEPTAR");
    botton e1(55,21,70,22,1,3,0,5,"AYUDA");
    d1.pon_fondo(1,3);
    d1.pon_marco();
    d1.pon_texto(13,8,1,17,15);
    e2.pon();
    e1.pon();
    e2.ilumina(3,15);
    d1.escribe(29,4,14,5,"PERFIL DE HABILIDADES");
    _setcursortype(_NOCURSOR);
    d1.lhor(28,5,51,1,1);
    d1.lver(51,4,4,1,1);
    register int i,j, pasa,xpos;
    typedef struct
    {
      char elemento[23];
      int puntos;
    } arreglo;
    arreglo c;
    arreglo habilidades[10];
    // vacia evhabi en el nuevo arreglo
    for(i=0;i<10;i++)
      habilidades[i].puntos=evhabi[i];
    // vaciar la clave de habilidades
    strcpy(habilidades[0].elemento,"SERVICIO SOCIAL");
    strcpy(habilidades[1].elemento,"EJECUTIVO-PERSUASIVO");
    strcpy(habilidades[2].elemento,"VERBAL");
    strcpy(habilidades[3].elemento,"ARTISTICO-PLASTICA");
```

```
strcpy(habilidades[4].elemento,"MUSICAL");
strcpy(habilidades[5].elemento,"ORGANIZACION");
strcpy(habilidades[6].elemento,"CIENTIFICA");
strcpy(habilidades[7].elemento,"CALCULO");
strcpy(habilidades[8].elemento,"MECANICA-CONSTRUCTIVA");
strcpy(habilidades[9].elemento,"DESTREZA MANUAL");
// ordenar el arreglo de habilidades, metodo de la burbuja
for(i=0;i<10;i++)
  for(j=9;j>i;j--)
       if (habilidades[j-1].puntos < habilidades[j].puntos)
       {
         c=habilidades[j-1];
         habilidades[j-1]=habilidades[j];
         habilidades[j]=c;
       }
d1.escribe(25,14,14,5,habilidades[0].elemento);
d1.escribe(25,16,14,5,habilidades[1].elemento);
d1.escribe(25,18,14,5,habilidades[2].elemento);
continua='1';
raton.inicia();
xpos=1;
do
{
  raton.activa();
  if(kbhit())
  {
       car=leetecla();
       if(car=='E')  // enter
       {
         if(xpos==1)              // aceptar
         {
           e2.push(3,15);
           delay(100);
           e2.pon();
           raton.desactiva();
           return(1);
         }
         if(xpos==2)              //ayuda
         {
           e1.push(3,15);
           delay(100);
           e1.pon();
        raton.desactiva();
        p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
           salva_pant(0,0,79,24,(char *) p);
        a1.pon_marco();
        a1.control(0);
        restaura_pant(0,0,79,24,(char *) p);
        free(p);
        e2.pon();
        e1.ilumina(3,15);
        raton.inicia();
         }
       }
       if((car=='C')&&(xpos==1))  // derecha
       {
```

```
            e2.pon();
            e1.ilumina(3,15);
            xpos=2;
          }
          if((car=='D')&&(xpos==2))   // izquierda
          {
            e1.pon();
            e2.ilumina(3,15);
            xpos=1;
          }
      }
      if(raton.boton==1)
      {
          if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))   // aceptar
          {
          e1.pon();
          e2.ilumina(3,15);
            e2.push(3,15);
            while(raton.boton==1)
              raton.activa();
            e2.pon();
            raton.desactiva();
            return(1);
          }
          if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))   // ayuda
          {
            e2.pon();
            e1.push(3,15);
            while(raton.boton==1)
              raton.activa();
            e1.pon();
            raton.desactiva();
        p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
            salva_pant(0,0,79,24,(char *) p);
          a1.pon_marco();
          a1.control(0);
          restaura_pant(0,0,79,24,(char *) p);
          free(p);
          e1.ilumina(3,15);
          raton.inicia();
          xpos=2;
          }
      }
    } while(continua=='1');
    return(1);
}


int almacenahabi(int evhabi[10])
{
    register int i;
    FILE *fp;
    if((fp=fopen("memoria.dat","a"))==NULL)   // a=adir datos en memoria
    {                                         // de trabajo
      error(1,"MEMORIA.DAT");
      return(0);
```

```
  }
  for(i=0;i<10;i++)
    fprintf(fp,"%d ",evhabi[i]);
  fprintf(fp,"\n");
  fclose(fp);
  return(1);

}



int main_habilidad()
{
  // evinter contiene las diez areas de habilidad y su puntuacion
  int evhabi[10];
  int habilidad [60];   // contiene las respuestas a las preguntas del cuestionario
  while(!(introducc1h()));
   while(!(introducc2h()))
     while(!(introducc1h()));
  cuest1h(habilidad);
  evalhabi(habilidad, evhabi);
  graficahabi(evhabi);
  ordenahabi(evhabi);
  almacenahabi(evhabi);
  return(1);
}


int resulta_habi()
{
  int evhabi[10];
  register int i;
  char c;
  FILE *fp;
  if((fp=fopen("memoria.dat","r"))==NULL)   // leer datos en memoria
  {                                         // de trabajo
    error(1,"MEMORIA.DAT");
    return(0);
  }
  // saltar un renglon
  do
    fscanf(fp,"%c",&c);
  while(c!='\n');
  for(i=0;i<10;i++)
    fscanf(fp,"%d ",&evhabi[i]);
  fscanf(fp,"\n");
  fclose(fp);
  graficahabi(evhabi);
  ordenahabi(evhabi);
  return(1);
}
```

```
// ARCHIVO CUES2.CPP.- Las funciones contenidas en este archivo
// son parte de la interface con el usuario y se encargan de
// aplicar el cuestionario de valores. Permiten al usuario
// introducir sus datos los cuales son almacenados en el
// archivo MEMORIA.DAT.
```

```
// Aplica el cuestionario de valores

#include "C:\TESIS\marco.h"

int cuest1v(char valor[22]);  // controla la aplicaci¢n del cuestionario
char lee_valor(pantalla c1, botton b1, botton b2,char valor[21],int pasada);
   // se encarga de leer los valores dados por el usuario
char cambia_pantav(pantalla c1, botton b1, botton b2, char valor[21], int pasada);
   // cambia las pantallas al aplicar el cuestionario
int introducc1v();    // despliega la primera introducci¢n
int introducc2v();    // despliega la segunda introducci¢n
int almacenavalor(char valor[22]);
// almacena los datos en memoria de trabajo, archivo memoria.dat



char lee_valor(pantalla c1, botton b1, botton b2,char valor[21],int pasada)
// regresa '0' en caso de p gina anterior y '1' en caso de aceptar
// boton b1 es cancelar, boton b2 es aceptar
{
  mouse raton;
  register int xpos, ypos,yposant, totpos, acpos;
  int valorpos;
  int num;
  char *fin, cancela, acepta;
  register int i;
  char car;
  ypos=10;        // posicion del cursor en pantalla, empieza en 10
  xpos=1;
  cancela='0';
  acepta='0';
  switch(pasada) {
    case 0: valorpos=0; break;
    case 1: valorpos=2; break;
    case 2: valorpos=5; break;
    case 3: valorpos=8; break;
    case 4: valorpos=11; break;
    case 5: valorpos=14; break;
    case 6: valorpos=16; break;
    case 7: valorpos=18; break;
    case 8: valorpos=20; break;
  }
  totpos=0;
  for(i=10;i<20;i++)
    if(que_hay(15,i+1)=='(')  // i+1 indica la posicion actual
    {
        gotoxy(16,i+1);
```

```
        putch(valor[valorpos+totpos]);
        totpos++;    // totpos es el nÉmero de elementos en pantalla
    }
  c1.pon_cursor(14,ypos,16,ypos,15,0);
  raton.inicia();
  do
  { raton.inicia();
    do
    {
        if(xpos==1)
          c1.pon_cursor(14,ypos,16,ypos,15,0);
        raton.activa();
        if(raton.boton==1)
        {
          if((raton.x>55)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))
          // se activ¢ boton cancela
          {
            b2.pon();
            b1.push(3,15);
            while(raton.boton==1)
                  raton.activa();
            b1.pon();
            cancela='1';
            c1.quita_cursor(14,ypos,16,ypos);
          }
          if((raton.x>35)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))
          // se activ¢ boton acepta
          {
            b1.pon();
            b2.push(3,15);
            while(raton.boton==1)
                  raton.activa();
            b2.pon();
            acepta='1';
            c1.quita_cursor(14,ypos,16,ypos);
          }
          if((raton.x>13)&&(raton.x<18)&&(raton.y>9)&&(raton.y<21))
            {
                  if(((que_hay(raton.x,raton.y) == '(')&&(que_hay(raton.x+2,raton.y)==')'))
                    ||((que_hay(raton.x,raton.y)==')')&&(que_hay(raton.x-2,raton.y)=='('))
                    ||((que_hay(raton.x-1,raton.y)=='(')&&(que_hay(raton.x+1,raton.y)==')')))
                  {
                    c1.quita_cursor(14,ypos,16,ypos);
                    ypos=raton.y - 1;
                    c1.pon_cursor(14,ypos,16,ypos,15,0);
                  }
            }
    } while((!kbhit())&&(acepta=='0')&&(cancela=='0'));
        raton.desactiva();
        if(kbhit())
        {
          car=leetecla();
          if(car=='H')   // barra espaciadora
          {
            if(xpos==1)
```

```
    {
        textbackground(0);
        gotoxy(16,ypos+1);
        // contar la posicion de la pregunta actual
        acpos=0;
        for(i=10;i<=ypos;i++)
        {
          if(que_hay(15,i+1)=='(')
              acpos++;
        }
        // meter al arreglo de datos
        if(valor[valorpos+acpos-1]==' ')
        {
          valor[valorpos+acpos-1]='þ';
          putch('þ');
        }
        else
        {
          valor[valorpos+acpos-1]=' ';
          putch(' ');
        }
    }
}
else
  switch(car)
  {
      case 'A':{  if(ypos==10)  // arriba
                      ypos=10;
                  else
                  if(xpos==1)
                  {
                        yposant=ypos;
                        while((que_hay(15,ypos)!='(')&&(ypos>11))
                          ypos--;
                        if(que_hay(15,ypos)=='(')
                        {
                          c1.quita_cursor(14,yposant,16,yposant);
                          ypos--;
                          c1.pon_cursor(14,ypos ,16,ypos,15,0);
                        }
                        else
                          ypos=yposant;
                  }
               }
               break;
      case 'B':{  // abajo
                        if(xpos==1)
                        {
                          yposant=ypos;
                          while((que_hay(15,ypos+2)!='(')&&(ypos<20))
                            ypos++;
                          if(que_hay(15,ypos+2)=='(')
                          {
                            c1.quita_cursor(14,yposant,16,yposant);
                            ypos++;
                            c1.pon_cursor(14,ypos,16,ypos,15,0);
```

```
                    }
                    else
                      ypos=yposant;
                  }
            }
            break;    //derecha
case 'C': {
            if(xpos==1)
            {
                yposant=ypos;
                while((que_hay(15,ypos+2)!='(')&&(ypos<20))
                  ypos++;
                if(que_hay(15,ypos+2)=='(')
                  ypos=yposant;
                else
                  {
                    xpos++;
                    c1.quita_cursor(14,yposant,16,yposant);
                    b2.ilumina(3,15);
                    ypos=yposant;
                  }
            }
            else
            if(xpos==2)
            {
                xpos++;
                b2.pon();
                b1.ilumina(3,15);
             }
            }
            break;
case 'D': {                 // izquierda
            if(xpos==2)
            {
                xpos--;
                b2.pon();
                c1.pon_cursor(14,ypos ,16,ypos,15,0);
            }
            else
                if(xpos==3)
                {
                  xpos--;
                  b1.pon();
                  b2.ilumina(3,15);
                }
            }
            break;
case 'E':{  if(xpos==2)//&&(ypos==8))     // enter
            {
                b2.push(3,15);
                delay(200);
                b2.pon();
                b2.ilumina(3,15);
                delay(100);
                acepta='1';
            }
```

```
                    else
                        if(xpos==3)//&&(ypos==8))
                        {
                          b1.push(3,15);
                          delay(200);
                          b1.pon();
                          b1.ilumina(3,15);
                          delay(100);
                          cancela='1';
                        }
                          else
                          {
                            yposant=ypos;
                            while((que_hay(15,ypos+2)!='(')&&(ypos<20))
                                ypos++;
                            if(que_hay(15,ypos+2)=='(')
                            {
                                c1.quita_cursor(14,yposant,16,yposant);
                                ypos++;
                                c1.pon_cursor(14,ypos,16,ypos,15,0);
                            }
                            else
                                {
                                  xpos++;
                                  c1.quita_cursor(14,yposant,16,yposant);
                                  b2.ilumina(3,15);
                                  ypos=yposant;
                                }
                          }
                    }
                    break;
            }
        }
} while((cancela=='0')&&(acepta=='0'));
  if(cancela=='1')
     return('0');
  else
     return('1');
}



char cambia_pantav(pantalla c1, botton b1, botton b2, char valor[21], int pasada)
{
  // desaparecer el cursor
  borra(7,9,73,19);
  c1.pon_texto(15,11,pasada*10 +1,10,15);
  b1.pon();
  b2.pon();
  cambia(14,10,18,21,7,5);
  return(lee_valor(c1,b1,b2,valor, pasada));
}
```

```
int cuest1v(char valor[22])
{
  register int i;
  pantalla c1(2,1,77,23,15,5,3,1,"VALOR.TXT");
  botton b2(35,21,50,22,1,3,0,5,"ADELANTE");
  botton b1(55,21,70,22,1,3,0,5,"P g.Ant.");
  clrscr();
  c1.pon_modo(3);
  c1.pon_fondo(1,3);
  c1.pon_grueso();
  c1.escribe(27,4,14,5,"    TABLA DE VALORES");
  c1.lhor(26,5,54,1,1);
  c1.escribe(8,7,14,5," Con la barra espaciadora marca aquellos valores que son");
  c1.escribe(8,8,14,5,"importantes para ti.");
  _setcursortype(_NOCURSOR);
  cambia(14,10,18,21,7,5);
  b1.pon();
  b2.pon();
  for(i=0;i<21;i++)     // limpia el arreglo de valores
    valor[i]=' ';
  valor[21]='\0';
  for(i=0;i<9;i++)
  {
        if(cambia_pantav(c1,b1,b2,valor,i)=='1'); // cambia pantalla
        else
          if(i==0)
            i=-1;
          else
            if(i>-1)
                i-=2;
  }
  return(1);
}


int introducc1v()
{
  int xpos;
  char car;
  char continua;
  mouse raton;
  _setcursortype(_NOCURSOR);
  pantalla a1(2,1,77,23,15,5,3,1,"int3v.txt");
  botton b2(35,21,50,22,1,3,0,5,"ADELANTE");
  botton b1(55,21,70,22,1,3,0,5,"P g.Ant.");
  a1.pon_fondo(1,3);
  a1.pon_marco();
  a1.pon_texto(14,9,1,17,15);
  b2.pon();
  b2.ilumina(3,15);
  b1.pon();
  a1.escribe(31,5,14,5,"MODULO VALORES");
  a1.lhor(28,6,48,1,1);
  a1.lver(48,5,5,1,1);
  continua='1';
  raton.inicia();
```

```
xpos=1;
do
{
  raton.activa();
  if(kbhit())
  {
      car=leetecla();
      if(car=='E')  // enter
      {
        if(xpos==1)
        {
          b2.push(3,15);
          delay(100);
          b2.pon();
          raton.desactiva();
          return(1);
        }
        if(xpos==2)
        {
          b1.push(3,15);
          delay(100);
          b1.pon();
          raton.desactiva();
          return(0);
        }
      }
      if((car=='C')&&(xpos==1))  // derecha
      {
        b2.pon();
        b1.ilumina(3,15);
        xpos=2;
      }
      if((car=='D')&&(xpos==2))  // izquierda
      {
        b1.pon();
        b2.ilumina(3,15);
        xpos=1;
      }
  }
  if(raton.boton==1)
  {
      if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
      {
        b2.push(3,15);
        while(raton.boton==1)
          raton.activa();
        b2.pon();
        raton.desactiva();
        return(1);
      }
      if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // p g. anterior
      {
        b2.pon();
        b1.push(3,15);
        while(raton.boton==1)
          raton.activa();
```

```
            b1.pon();
            raton.desactiva();
            return(0);
          }
      }
    } while(continua=='1');
    return(0);
}


int introducc2v()
{
    int xpos;
    char car;
    mouse raton;
    char continua;
    _setcursortype(_NOCURSOR);
    pantalla d1(2,1,77,23,15,5,3,1,"int31v.txt");
    botton e2(35,21,50,22,1,3,0,5,"ADELANTE");
    botton e1(55,21,70,22,1,3,0,5,"P g.Ant.");
    d1.pon_fondo(1,3);
    d1.pon_marco();
    d1.pon_texto(14,10,1,17,15);
    e2.pon();
    e2.ilumina(3,15);
    e1.pon();
    d1.escribe(29,4,14,5,"  MODULO VALORES");
    d1.lhor(28,5,48,1,1);
    d1.lver(48,4,4,1,1);
    continua='1';
    raton.inicia();
    xpos=1;
    do
    {
      raton.activa();
      if(kbhit())
      {
          car=leetecla();
          if(car=='E')  // enter
          {
            if(xpos==1)
            {
              e2.push(3,15);
              delay(100);
              e2.pon();
              raton.desactiva();
              return(1);
            }
            if(xpos==2)
            {
              e1.push(3,15);
              delay(100);
              e1.pon();
              raton.desactiva();
              return(0);
            }
```

```
        }
        if((car=='C')&&(xpos==1))  // derecha
        {
          e2.pon();
          e1.ilumina(3,15);
          xpos=2;
        }
        if((car=='D')&&(xpos==2))  // izquierda
        {
          e1.pon();
          e2.ilumina(3,15);
          xpos=1;
        }
    }
    if(raton.boton==1)
    {
        if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
        {
          e2.push(3,15);
          while(raton.boton==1)
            raton.activa();
          e2.pon();
          raton.desactiva();
          return(1);
        }
        if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // p g. anterior
        {
          e2.pon();
          e1.push(3,15);
          while(raton.boton==1)
            raton.activa();
          e1.pon();
          raton.desactiva();
          return(0);
        }
    }
  } while(continua=='1');
  return(0);
}


int almacenavalor(char valor[22])
{
   register int i;
   FILE *fp;
   if((fp=fopen("memoria.dat","a"))==NULL)  // a=adir datos en memoria
   {                                        // de trabajo
     error(1,"MEMORIA.DAT");
     return(0);
   }
   for(i=0;i<22;i++)
     fprintf(fp,"%c",valor[i]);
   fprintf(fp,"\n");
   fclose(fp);
   return(1);
}
```

```
int main_valor()
{
  char valor[22];
  while(!(introducc1v()));
  while(!(introducc2v()))
    while(!(introducc1v()));
  cuest1v(valor);
  almacenavalor(valor);
  return(1);
}
```

```
// ARCHIVO CUES3.CPP.- Las funciones que contiene son parte de
// la interface con el usuario. Especificamente su labor es
// aplicar el cuestionario de ocupaciones. Los datos son
// almacenados en el archivo memoria.dat

// Aplica el cuestionario de ocupaciones

#include "C:\TESIS\marco.h"

void cuest1o(char ocupa[59]);  // controla la aplicacion del cuestionario
char lee_ocupa(pantalla c1, botton b1, botton b2,char ocupa[59],int pasada);
   // se encarga de leer los valores del cuestionario
char cambia_pantao(pantalla c1, botton b1, botton b2, char ocupa[59], int pasada);
   // cambia la pantalla al aplicar el cuestionario
int introducc1o(char archi[12]);    // despliega las introducciones
int almacenaocupa(char ocupa[59]);  // almacena los resultados



char lee_ocupa(pantalla c1, botton b1, botton b2,char ocupa[59],int pasada)
// regresa '0' en caso de cancelar y '1' en caso de aceptar
// boton b1 es cancelar, boton b2 es aceptar
{
  mouse raton;
  register int xpos, ypos,yposant, acpos;
  static int posant; //numero de elementos anteriores en el arreglo
  register int totpos;  // numero de elementos en la pantalla actual
  register int posact;  // elemento actual del cursor en la pantalla
  int num;
  char *fin, cancela, acepta;
  register int i;
  char car;
  ypos=0;
  do
    ypos++;
  while(que_hay(10,ypos+1)!='('); // posicion del cursor en pantalla, empieza en 13
  xpos=1;
  cancela='0';
  acepta='0';
  switch(pasada) {
    case 0: posant=0; break;
    case 1: posant=4; break;
    case 2: posant=7; break;
    case 3: posant=13; break;
    case 4: posant=17; break;
    case 5: posant=19; break;
    case 6: posant=21; break;
    case 7: posant=24; break;
    case 8: posant=25; break;
    case 9: posant=27; break;
    case 10:posant=29; break;
    case 11:posant=30; break;
    case 12:posant=32; break;
    case 13:posant=35; break;
    case 14:posant=38; break;
```

```
    case 15:posant=44; break;
    case 16:posant=48; break;
    case 17:posant=52; break;
    case 18:posant=53; break;
    case 19:posant=55; break;
    case 20:posant=56; break;
  }
  totpos=0;
  for(i=10;i<20;i++)
    if(que_hay(10,i+1)=='(')   // i+1 indica la posicion actual
    {
        gotoxy(11,i+1);
        putch(ocupa[posant+totpos]);
        totpos++;    // totpos es el nÉmero de elementos en pantalla
    }
  c1.pon_cursor(9,ypos,11,ypos,15,0);
  raton.inicia();
  do
  { raton.inicia();
    do
    {
        if(xpos==1)
          c1.pon_cursor(9,ypos,11,ypos,15,0);
        raton.activa();
        if(raton.boton==1)
        {
          if(((raton.x>55)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))
          // se activ¢ boton cancela
          {
            b2.pon();
            b1.push(3,15);
            while(raton.boton==1)
                raton.activa();
            b1.pon();
            cancela='1';
            c1.quita_cursor(9,ypos,11,ypos);
          }
          if((raton.x>35)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))
          // se activ¢ boton acepta
          {
            b1.pon();
            b2.push(3,15);
            while(raton.boton==1)
                raton.activa();
            b2.pon();
            acepta='1';
            c1.quita_cursor(9,ypos,11,ypos);
          }
          if((raton.x>9)&&(raton.x<13)&&(raton.y>9)&&(raton.y<21))
            {
                if(((que_hay(raton.x,raton.y) == '(')&&(que_hay(raton.x+2,raton.y)==')'))
                    ||((que_hay(raton.x,raton.y)==')')&&(que_hay(raton.x-2,raton.y)=='('))
                    ||((que_hay(raton.x-1,raton.y)=='(')&&(que_hay(raton.x+1,raton.y)==')')))
                {
                  c1.quita_cursor(9,ypos,11,ypos);
                  ypos=raton.y - 1;
```

```
                c1.pon_cursor(9,ypos,11,ypos,15,0);
              }
          }
      }
} while((!kbhit())&&(acepta=='0')&&(cancela=='0'));
      raton.desactiva();
      if(kbhit())
      {
        car=leetecla();
        if(car=='H')   // barra espaciadora
        {
          if(xpos==1)
          {
              textbackground(0);
              gotoxy(11,ypos+1);
              // contar la posicion de la pregunta actual
              posact=0;
              for(i=5;i<=ypos;i++)
              {
                if(que_hay(10,i+1)=='(')
                    posact++;
              }
              // meter al arreglo de datos
              if(ocupa[posact+posant-1]==' ')
              {
                ocupa[posact+posant-1]='b';
                putch('b');
              }
              else
              {
                ocupa[posact+posant-1]=' ';
                putch(' ');
              }
          }
        }
        else
          switch(car)
          {
              case 'A':{  if(ypos==10)   // arriba
                            ypos=10;
                          else
                          if(xpos==1)
                          {
                              yposant=ypos;
                              while((que_hay(10,ypos)!='(')&&(ypos>5))
                                ypos--;
                              if(que_hay(10,ypos)=='(')
                              {
                                c1.quita_cursor(9,yposant,11,yposant);
                                ypos--;
                                c1.pon_cursor(9,ypos ,11,ypos,15,0);
                              }
                              else
                                ypos=yposant;
                          }
                       }
```

```
            break;
case 'B':{  // abajo
                    if(xpos==1)
                    {
                      yposant=ypos;
                      while((que_hay(10,ypos+2)!='()&&(ypos<20))
                        ypos++;
                      if(que_hay(10,ypos+2)=='()
                      {
                        c1.quita_cursor(9,yposant,11,yposant);
                        ypos++;
                        c1.pon_cursor(9,ypos,11,ypos,15,0);
                      }
                      else
                        ypos=yposant;
                    }
            }
            break;    //derecha
case 'C': {
              if(xpos==1)
              {
                    yposant=ypos;
                    while((que_hay(10,ypos+2)!='()&&(ypos<20))
                      ypos++;
                    if(que_hay(15,ypos+2)=='()
                      ypos=yposant;
                    else
                      {
                        xpos++;
                        c1.quita_cursor(9,yposant,11,yposant);
                        b2.ilumina(3,15);
                        ypos=yposant;
                      }
              }
              else
              if(xpos==2)
              {
                    xpos++;
                    b2.pon();
                    b1.ilumina(3,15);
               }
            }
            break;
case 'D': {                 // izquierda
              if(xpos==2)
              {
                    xpos--;
                    b2.pon();
                    c1.pon_cursor(9,ypos ,11,ypos,15,0);
              }
              else
                    if(xpos==3)
                    {
                      xpos--;
                      b1.pon();
                      b2.ilumina(3,15);
```

```
                    }
                }
                break;
        case 'E':{  if(xpos==2)//&&(ypos==8))    // enter
                    {
                        b2.push(3,15);
                        delay(200);
                        b2.pon();
                        b2.ilumina(3,15);
                        delay(100);
                        acepta='1';
                    }
                    else
                        if(xpos==3)//&&(ypos==8))
                        {
                          b1.push(3,15);
                          delay(200);
                          b1.pon();
                          b1.ilumina(3,15);
                          delay(100);
                          cancela='1';
                        }
                          else
                          {
                            yposant=ypos;
                            while((que_hay(10,ypos+2)!='(')&&(ypos<20))
                                ypos++;
                            if(que_hay(10,ypos+2)=='(')
                            {
                                c1.quita_cursor(9,yposant,11,yposant);
                                ypos++;
                                c1.pon_cursor(9,ypos,11,ypos,15,0);
                            }
                            else
                                {
                                  xpos++;
                                  c1.quita_cursor(9,yposant,11,yposant);
                                  b2.ilumina(3,15);
                                  ypos=yposant;
                                }
                          }
                    }
                    break;
            }
        }
} while((cancela=='0')&&(acepta=='0'));
  if(cancela=='1')
    return('0');
  else
    return('1');
}



char cambia_pantao(pantalla c1, botton b1, botton b2, char ocupa[59], int pasada)
```

```
{
  // desaparecer el cursor
  borra(7,9,73,19);
  c1.pon_texto(10,11,pasada*10 +1,10,15);
  b1.pon();
  b2.pon();
  cambia(7,10,11,21,7,5);
  return(lee_ocupa(c1,b1,b2,ocupa, pasada));
}



void cuest1o(char ocupa[59])
{
  register int i;
  static pantalla c1(2,1,77,23,15,5,3,1,"OCUPA.TXT");
  static botton b2(35,21,50,22,1,3,0,5,"ADELANTE");
  static botton b1(55,21,70,22,1,3,0,5,"P g.Ant.");
  clrscr();
  c1.pon_modo(3);
  c1.pon_fondo(1,3);
  c1.pon_grueso();
  c1.pon_texto(10,11,1,10,15);
  c1.escribe(26,4,14,5,"EL MUNDO DE LAS OCUPACIONES");
  c1.lhor(23,5,54,1,1);
  c1.escribe(8,6,14,5,"  Con la barra espaciadora marca aquellos campos ocupacionales");
  c1.escribe(8,7,14,5,"que te interesen de acuerdo con la  satisfacci¢n  personal,  la");
  c1.escribe(8,8,14,5,"remuneraci¢n econ¢mica y el papel que desempe¤a en la sociedad.");
  _setcursortype(_NOCURSOR);
  cambia(7,10,11,21,7,5);
  b1.pon();
  b2.pon();
  for(i=0;i<58;i++)     // limpia el arreglo de ocupaciones
    ocupa[i]=' ';
  ocupa[58]='\0';
  for(i=0;i<21;i++)
  {
        if(cambia_pantao(c1,b1,b2,ocupa,i)=='1');  // cambia pantalla
        else
          if(i==0)
            i=-1;
          else
            if(i>-1)
                i-=2;
  }
}


int introducc1o(char archi[11])
{
  int xpos;
  char car;
  char continua;
  mouse raton;
  _setcursortype(_NOCURSOR);
  pantalla a1(2,1,77,23,15,5,3,1,archi);
```

```
static botton b2(35,21,50,22,1,3,0,5,"ADELANTE");
static botton b1(55,21,70,22,1,3,0,5,"P g.Ant.");
a1.pon_fondo(1,3);
a1.pon_marco();
b1.pon();
b2.pon();
b2.ilumina(3,15);
b1.pon();
a1.escribe(26,4,14,5,"EL MUNDO DE LAS OCUPACIONES");
a1.lhor(24,5,55,1,1);
a1.lver(55,4,4,1,1);
continua='1';
raton.inicia();
xpos=1;
a1.pon_texto(11,7,1,17,15);
do
{
  raton.activa();
  if(kbhit())
  {
       car=leetecla();
       if(car=='E')  // enter
       {
         if(xpos==1)
         {
           b2.push(3,15);
           delay(100);
           b2.pon();
           raton.desactiva();
           return(1);
         }
         if(xpos==2)
         {
           b1.push(3,15);
           delay(100);
           b1.pon();
           raton.desactiva();
           return(0);
         }
       }
       if((car=='C')&&(xpos==1))  // derecha
       {
         b2.pon();
         b1.ilumina(3,15);
         xpos=2;
       }
       if((car=='D')&&(xpos==2))  // izquierda
       {
         b1.pon();
         b2.ilumina(3,15);
         xpos=1;
       }
  }
  if(raton.boton==1)
  {
       if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
```

```
        {
          b2.push(3,15);
          while(raton.boton==1)
            raton.activa();
          b2.pon();
          raton.desactiva();
          return(1);
        }
        if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // cancelar
        {
          b2.pon();
          b1.push(3,15);
          while(raton.boton==1)
            raton.activa();
          b1.pon();
          raton.desactiva();
          return(0);
        }
    }
  } while(continua=='1');
  return(0);
}


int almacenaocupa(char ocupa[59])
{
   register int i;
   FILE *fp;
   if((fp=fopen("memoria.dat","a"))==NULL)  // a=adir datos en memoria
   {                                       // de trabajo
     error(1,"MEMORIA.DAT");
     return(0);
   }
   for(i=0;i<59;i++)
     fprintf(fp,"%c",ocupa[i]);
   fprintf(fp,"\n");
   fclose(fp);
   return(1);
}




int main_ocupa()
{
   char ocupa[59];
   while(!(introducc1o("int3oc.txt")));

   while(!(introducc1o("int3oc1.txt")))
     while(!(introducc1o("int3oc.txt")));

   while(!(introducc1o("int3oc2.txt")))
```

```
    while(!(introducc1o("int3oc1.txt")))
        while(!(introducc1o("int3oc.txt")));

  while(!(introducc1o("int3oc3.txt")))
    while(!(introducc1o("int3oc2.txt")))
        while(!(introducc1o("int3oc1.txt")))
          while(!(introducc1o("int3oc.txt")));

  while(!(introducc1o("int3oc4.txt")))
    while(!(introducc1o("int3oc3.txt")))
        while(!(introducc1o("int3oc2.txt")))
          while(!(introducc1o("int3oc1.txt")))
            while(!(introducc1o("int3oc.txt")));

  while(!(introducc1o("int3oc5.txt")))
    while(!(introducc1o("int3oc4.txt")))
        while(!(introducc1o("int3oc3.txt")))
          while(!(introducc1o("int3oc2.txt")))
            while(!(introducc1o("int3oc1.txt")))
                while(!(introducc1o("int3oc.txt")));

  while(!(introducc1o("int3oc6.txt")))
    while(!(introducc1o("int3oc5.txt")))
        while(!(introducc1o("int3oc4.txt")))
          while(!(introducc1o("int3oc3.txt")))
            while(!(introducc1o("int3oc2.txt")))
                while(!(introducc1o("int3oc1.txt")))
                  while(!(introducc1o("int3oc.txt")));
  cuest1o(ocupa);
  almacenaocupa(ocupa);
  return(1);
}
```

```
// ARCHIVO CUES4.CPP.- Las funciones que contiene son parte de
// la interface con el usuario . Se encarga de aplicar el
// cuestionario de reas de estudio profesional. Almacena los
// datos en el archivo MEMORIA.DAT.
```

```
// Aplica el cuestionario de reas de estudio profesional

#include "C:\TESIS\marco.h"

void cuest1ae(char area[41]);  // controla la aplicacion del cuestionario
char lee_area(pantalla c1, botton b1, botton b2,char area[41],int pasada);
  // lee los datos del usuario
char cambia_pantaae(pantalla c1, botton b1, botton b2, char area[41], int pasada);
  // cambia la pantalla en la aplicacion del cuestionario
int introducc1ae(char archi[12]);    // despliega las introducciones
int almacenaae(char area[41]);       // almacena los resultados


char lee_area(pantalla c1, botton b1, botton b2,char area[41],int pasada)
// regresa '0' en caso de pagina anterior y '1' en caso de aceptar
// boton b1 es pagina anterior, boton b2 es aceptar
{
  mouse raton;
  register int xpos, ypos,yposant, acpos;
  static int posant;  //numero de elementos anteriores en el arreglo
  register int totpos;  // numero de elementos en la pantalla actual
  register int posact;  // elemento actual del cursor en la pantalla
  int num;
  char *fin, cancela, acepta;
  register int i;
  char car;
  ypos=0;
  do
    ypos++;
  while(que_hay(10,ypos+1)!='('); // posicion del cursor en pantalla, empieza en 13
  xpos=1;
  cancela='0';
  acepta='0';
  switch(pasada) {
    case 0: posant=0; break;
    case 1: posant=2; break;
    case 2: posant=4; break;
    case 3: posant=6; break;
    case 4: posant=8; break;
    case 5: posant=10; break;
    case 6: posant=12; break;
    case 7: posant=14; break;
    case 8: posant=16; break;
    case 9: posant=17; break;
    case 10:posant=18; break;
    case 11:posant=20; break;
```

```
  case 12:posant=22; break;
  case 13:posant=23; break;
  case 14:posant=25; break;
  case 15:posant=26; break;
  case 16:posant=27; break;
  case 17:posant=28; break;
  case 18:posant=30; break;
  case 19:posant=32; break;
  case 20:posant=34; break;
  case 21:posant=36; break;
  case 22:posant=37; break;
  case 23:posant=39; break;
}
totpos=0;
for(i=10;i<20;i++)
  if(que_hay(10,i+1)=='(')  // i+1 indica la posicion actual
  {
       gotoxy(11,i+1);
       putch(area[posant+totpos]);
       totpos++;   // totpos es el n£mero de elementos en pantalla
  }
c1.pon_cursor(9,ypos,11,ypos,15,0);
raton.inicia();
do
{ raton.inicia();
 do
  {
       if(xpos==1)
         c1.pon_cursor(9,ypos,11,ypos,15,0);
       raton.activa();
       if(raton.boton==1)
       {
         if((raton.x>55)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))
         // se activ¢ boton pag. anterior
         {
           b2.pon();
           b1.push(3,15);
           while(raton.boton==1)
                raton.activa();
           b1.pon();
           cancela='1';
           c1.quita_cursor(9,ypos,11,ypos);
           b1.pon();
           b2.pon();
         }
         if((raton.x>35)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))
         // se activ¢ boton acepta
         {
           b1.pon();
           b2.push(3,15);
           while(raton.boton==1)
                raton.activa();
           b2.pon();
           acepta='1';
           c1.quita_cursor(9,ypos,11,ypos);
         }
```

```
        if((raton.x>9)&&(raton.x<13)&&(raton.y>9)&&(raton.y<21))
          {
              if(((que_hay(raton.x,raton.y) == '(')&&(que_hay(raton.x+2,raton.y)==')')
                 ||((que_hay(raton.x,raton.y)==')')&&(que_hay(raton.x-2,raton.y)=='('))
                 ||((que_hay(raton.x-1,raton.y)=='(')&&(que_hay(raton.x+1,raton.y)==')')))
              {
                c1.quita_cursor(9,ypos,11,ypos);
                ypos=raton.y - 1;
                c1.pon_cursor(9,ypos,11,ypos,15,0);
              }
          }
      }
  } while((!kbhit())&&(acepta=='0')&&(cancela=='0'));
      raton.desactiva();
      if(kbhit())
      {
        car=leetecla();
        if(car=='H')   // barra espaciadora
        {
          if(xpos==1)
          {
              textbackground(0);
              gotoxy(11,ypos+1);
              // contar la posicion de la pregunta actual
              posact=0;
              for(i=5;i<=ypos;i++)
              {
                if(que_hay(10,i+1)=='(')
                    posact++;
              }
              // meter al arreglo de datos
              if(area[posact+posant-1]==' ')
              {
                area[posact+posant-1]='p';
                putch('p');
              }
              else
              {
                area[posact+posant-1]=' ';
                putch(' ');
              }
          }
        }
        else
          switch(car)
          {
              case 'A':{  if(ypos==10)   // arriba
                            ypos=10;
                          else
                          if(xpos==1)
                          {
                              yposant=ypos;
                              while((que_hay(10,ypos)!='(')&&(ypos>5))
                                ypos--;
                              if(que_hay(10,ypos)=='(')
                              {
```

```
                    c1.quita_cursor(9,yposant,11,yposant);
                    ypos--;
                    c1.pon_cursor(9,ypos ,11,ypos,15,0);
                  }
                  else
                    ypos=yposant;
              }
            }
            break;
case 'B':{  // abajo
                  if(xpos==1)
                  {
                    yposant=ypos;
                    while((que_hay(10,ypos+2)!='(')&&(ypos<20))
                      ypos++;
                    if(que_hay(10,ypos+2)=='(')
                    {
                      c1.quita_cursor(9,yposant,11,yposant);
                      ypos++;
                      c1.pon_cursor(9,ypos,11,ypos,15,0);
                    }
                    else
                      ypos=yposant;
                  }
            }
            break;    //derecha
case 'C': {
              if(xpos==1)
              {
                    yposant=ypos;
                    while((que_hay(10,ypos+2)!='(')&&(ypos<20))
                      ypos++;
                    if(que_hay(15,ypos+2)=='(')
                      ypos=yposant;
                    else
                     {
                       xpos++;
                       c1.quita_cursor(9,yposant,11,yposant);
                       b2.ilumina(3,15);
                       ypos=yposant;
                     }
              }
              else
              if(xpos==2)
              {
                    xpos++;
                    b2.pon();
                    b1.ilumina(3,15);
              }
            }
            break;
case 'D': {           // Izquierda
              if(xpos==2)
              {
                    xpos--;
                    b2.pon();
```

```
                    c1.pon_cursor(9,ypos ,11,ypos,15,0);
                }
                else
                    if(xpos==3)
                    {
                      xpos--;
                      b1.pon();
                      b2.ilumina(3,15);
                    }
                }
                break;
        case 'E':{  if(xpos==2)    // enter
                    {
                      b2.push(3,15);
                      delay(200);
                      b2.pon();
                      b2.ilumina(3,15);
                      delay(100);
                      acepta='1';
                    }
                    else
                      if(xpos==3)
                      {
                        b1.push(3,15);
                        delay(200);
                        b1.pon();
                        b1.ilumina(3,15);
                        delay(100);
                        cancela='1';
                      }
                        else
                        {
                          yposant=ypos;
                          while((que_hay(10,ypos+2)!='(')&&(ypos<20))
                              ypos++;
                          if(que_hay(10,ypos+2)=='(')
                          {
                              c1.quita_cursor(9,yposant,11,yposant);
                              ypos++;
                              c1.pon_cursor(9,ypos,11,ypos,15,0);
                          }
                          else
                              {
                                xpos++;
                                c1.quita_cursor(9,yposant,11,yposant);
                                b2.ilumina(3,15);
                                ypos=yposant;
                              }
                        }
                    }
                    break;
            }
        }
} while((cancela=='0')&&(acepta=='0'));
  b1.pon();
  b2.pon();
```

```
  if(cancela=='1')
    return('0');
  else
    return('1');
}



char cambia_pantaae(pantalla c1, botton b1, botton b2, char area[41], int pasada)
{
  // desaparecer el cursor
  borra(7,9,73,19);
  c1.pon_texto(10,11,pasada*10 +1,10,15);
  b1.pon();
  b2.pon();
  cambia(7,10,11,21,7,5);
  return(lee_area(c1,b1,b2,area, pasada));
}



void cuest1ae(char area[41])
{
  register int i;
  static pantalla c1(2,1,77,23,15,5,3,1,"AREA.TXT");
  static botton b2(35,21,50,22,1,3,0,5,"ADELANTE");
  static botton b1(55,21,70,22,1,3,0,5,"P g.Ant.");
  clrscr();
  c1.pon_modo(3);
  c1.pon_fondo(1,3);
  c1.pon_grueso();
  c1.pon_texto(10,11,1,10,15);
  c1.escribe(25,4,14,5,"AREAS DE ESTUDIO PROFESIONAL");
  c1.lhor(22,5,54,1,1);
  c1.escribe(10,6,14,5,"  Con la  barra  espaciadora  marca  aquellas profesiones que");
  c1.escribe(10,7,14,5,"consideres m s apropiadas  para  ti  tomando en  cuenta  tus");
  c1.escribe(10,8,14,5,"antecedentes acad,micos.");
  _setcursortype(_NOCURSOR);
  cambia(7,10,11,21,7,5);
  b1.pon();
  b2.pon();
  for(i=0;i<40;i++)      // limpia el arreglo de ocupaciones
    area[i]=' ';
  area[40]='\0';
  for(i=0;i<24;i++)
  {
        if(cambia_pantaae(c1,b1,b2,area,i)=='1'); // cambia pantalla
        else
          if(i==0)
            i=-1;
          else
            if(i>-1)
                i-=2;
  }
}
```

```
int introducc1ae(char archi[11])
{
  int xpos;
  char car;
  char continua;
  mouse raton;
  _setcursortype(_NOCURSOR);
  pantalla a1(2,1,77,23,15,5,3,1,archi);
  static botton b2(35,21,50,22,1,3,0,5,"ADELANTE");
  static botton b1(55,21,70,22,1,3,0,5,"P g.Ant.");
  a1.pon_fondo(1,3);
  a1.pon_marco();
  b2.pon();
  b2.ilumina(3,15);
  b1.pon();
  if((strcmpi(archi,"int3ae1.txt")))
  {
    a1.escribe(25,4,14,5,"AREAS DE ESTUDIO PROFESIONAL");
    a1.lhor(23,5,55,1,1);
    a1.lver(55,4,4,1,1);
  }
  continua='1';
  raton.inicia();
  xpos=1;
  if((strcmpi(archi,"int3ae1.txt")))
    a1.pon_texto(9,7,1,17,15);
  else
     a1.pon_texto(9,4,1,17,15);
  do
  {
    raton.activa();
    if(kbhit())
    {
          car=leetecla();
          if(car=='E')  // enter
          {
            if(xpos==1)
            {
              b2.push(3,15);
              delay(100);
              b2.pon();
              raton.desactiva();
              return(1);
            }
            if(xpos==2)
            {
              b1.push(3,15);
              delay(100);
              b1.pon();
              raton.desactiva();
              return(0);
            }
          }
          if((car=='C')&&(xpos==1))  // derecha
          {
            b2.pon();
```

```
            b1.ilumina(3,15);
            xpos=2;
          }
          if((car=='D')&&(xpos==2))  // izquierda
          {
            b1.pon();
            b2.ilumina(3,15);
            xpos=1;
          }
      }
      if(raton.boton==1)
      {
          if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
          {
            b2.push(3,15);
            while(raton.boton==1)
              raton.activa();
            b2.pon();
            raton.desactiva();
            return(1);
          }
          if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // cancelar
          {
            b2.pon();
            b1.push(3,15);
            while(raton.boton==1)
              raton.activa();
            b1.pon();
            raton.desactiva();
            return(0);
          }
      }
    } while(continua=='1');
    return(0);
}


int almacenaae(char area[41])
{
    register int i;
    FILE *fp;
    if((fp=fopen("memoria.dat","a"))==NULL)  // a=adir datos en memoria
    {                                        // de trabajo
      error(1,"MEMORIA.DAT");
      return(0);
    }
    for(i=0;i<41;i++)
      fprintf(fp,"%c",area[i]);
    fprintf(fp,"\n");
    fclose(fp);
    return(1);
}

int main_area()
{
    char area[41];
```

```
  while(!(introducc1ae("int3ae.txt")))
    ;
  while(!(introducc1ae("int3ae2.txt")))
    while(!(introducc1ae("int3ae.txt")));
  cuest1ae(area);
  almacenaae(area);
  return(1);
}
```

```
// ARCHIVO RESULTA.CPP.- Es parte de la interface de usuario y
// contiene las funciones que realizan las graficas del perfil
// interes-habilidad y el perfil de personalidad.

// Realiza las graficas interes-habilidad y la de personalidad
// Es parte de la interface de usuario

#include "c:\tesis\marco.h"

// Las sig. funciones estan contenidas en este archivo aunque
// no son ejecutadas desde este.

//int grafica_inter_y_habi();
// muestra la grafica interes-habilidad
//int graficaper(int valorper[6]);
// muestra la grafica de los tipos de personalidad
//int main_resulta();
// controla todo el proceso
//int pon_mejorih(char mejorih[4]);
// despliega los tres mejores pares interes-habilidad

int introduccres(char archi[9]);
// despliega la introduccion a los tipos de personalidad



int introduccres(char archi[9])
{
  int xpos;
  char car;
  char continua;
  mouse raton;
  _setcursortype(_NOCURSOR);
  pantalla a1(2,1,77,23,15,5,3,1,archi);
  botton b2(35,21,50,22,1,3,0,5,"ADELANTE");
  botton b1(55,21,70,22,1,3,0,5,"P g.Ant.");
  a1.pon_fondo(1,3);
  a1.pon_marco();
  b1.pon();
  b2.pon();
  b2.ilumina(3,15);
  b1.pon();
  a1.escribe(32,4,14,5,"TU PERSONALIDAD");
  a1.lhor(30,5,49,1,1);
  a1.lver(49,4,4,1,1);
  continua='1';
  raton.inicia();
  xpos=1;
  a1.pon_texto(11,7,1,17,15);
  do
  {
    raton.activa();
    if(kbhit())
    {
```

```
        car=leetecla();
        if(car=='E')  // enter
        {
          if(xpos==1)
          {
            b2.push(3,15);
            delay(100);
            b2.pon();
            raton.desactiva();
            return(1);
          }
          if(xpos==2)
          {
            b1.push(3,15);
            delay(100);
            b1.pon();
            raton.desactiva();
            return(0);
          }
        }
        if((car=='C')&&(xpos==1))  // derecha
        {
          b2.pon();
          b1.ilumina(3,15);
          xpos=2;
        }
        if((car=='D')&&(xpos==2))  // izquierda
        {
          b1.pon();
          b2.ilumina(3,15);
          xpos=1;
        }
    }
    if(raton.boton==1)
    {
        if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
        {
          b2.push(3,15);
          while(raton.boton==1)
            raton.activa();
          b2.pon();
          raton.desactiva();
          return(1);
        }
        if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // cancelar
        {
          b2.pon();
          b1.push(3,15);
          while(raton.boton==1)
            raton.activa();
          b1.pon();
          raton.desactiva();
          return(0);
        }
    }
  } while(continua=='1');
```

```
  return(0);
}



int grafica_inter_y_habi()
// hace la grafica conjunta de interes-habilidad
{
  unsigned char *p;
  pantemerge a1(15,4,65,21,15,7,0,8,"ayuint.txt");
  int evinter[10];
  int evhabi[10];
  mouse raton;
  char continua, car;
  div_t x;
  FILE *fp;
  register int i, j, por, cuadros, ypos;
  pantalla c1(0,0,80,25,5,5,5,5,"          ");
  botton b2(61,18,76,19,1,3,0,5,"ACEPTAR");
  botton b1(61,21,76,22,1,3,0,5," AYUDA");
  c1.pon_marco();
  b2.pon();
  b2.ilumina(3,15);
  b1.pon();
  c1.escribe(29,1,14,5,"GRAFICA INTERES-HABILIDAD");
  c1.escribe(2,4,15,5, "ASISTENCIAL");
  c1.escribe(2,8,15,5, "EJECUTIVO-PERSUASIVO");
  c1.escribe(2,8,15,5, "VERBAL");
  c1.escribe(2,10,15,5, "ARTISTICO-PLASTICO");
  c1.escribe(2,12,15,5,"MUSICAL");
  c1.escribe(2,14,15,5,"ORGANIZACION");
  c1.escribe(2,16,15,5,"CIENTIFICO");
  c1.escribe(2,18,15,5,"CALCULO");
  c1.escribe(2,20,15,5,"MECANICO-CONSTRUCT.");
  c1.escribe(2,22,15,5,"ACT. AL AIRE LIBRE");
  c1.escribe(22,25,15,5,"  10 20 30 40 50 60 70 80 90 100     %");
  c1.escribe(61,4,1,5,"ÛÛ");
  c1.escribe(64,4,14,5,"= intereses");
  c1.escribe(61,6,9,5,"ÛÛ");
  c1.escribe(64,6,14,5,"= habilidades");
  c1.lhor(23,24,61,15,0);
  c1.lver(23,3,23,15,0);
  _setcursortype(_NOCURSOR);
  gotoxy(23,24);  putch('À');
  // leer evinter de archivo memoria.dat
  if((fp=fopen("memoria.dat","r"))==NULL)
  {
    error(1,"MEMORIA.DAT");
    return(0);
  }
  for(i=0;i<10;i++)
    fscanf(fp,"%d ",&evinter[i]);
  fscanf(fp,"\n");
  // leer evhabi del archivo memoria.dat
  for(i=0;i<10;i++)
```

```
  fscanf(fp,"%d ",&evhabi[i]);
fscanf(fp,"\n");
fclose(fp);
for(i=0;i<10;i++)
{
  x=div(evinter[i]*100,24);
  por=x.quot;
  x=div(por*30,100);
  cuadros=x.quot;
  for(j=0;j<cuadros;j++)
       c1.escribe(24+j,4+i*2,1,5,"Ü");
  x=div(evhabi[i]*100,24);
  por=x.quot;
  x=div(por*30,100);
  cuadros=x.quot;
  for(j=0;j<cuadros;j++)
       c1.escribe(24+j,5+i*2,9,1,"Ü");
}
continua='1';
raton.inicia();
ypos=1;
do
{
  raton.activa();
  if(kbhit())
  {
       car=leetecla();
       if(car=='E')  // enter       // aceptar
       {
         if(ypos==1)
         {
        b1.pon();
          b2.push(3,15);
          delay(100);
          b2.pon();
          raton.desactiva();
          return(1);
         }
         if(ypos==2)                 // ayuda
         {
          b1.push(3,15);
          delay(100);
          b1.pon();
        raton.desactiva();
        p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
          salva_pant(0,0,79,24,(char *) p);
        a1.pon_marco();
        a1.control(0);
        restaura_pant(0,0,79,24,(char *) p);
        free(p);
        b1.ilumina(3,15);
        b2.pon();
        raton.inicia();
         }
       }
       if((car=='A')&&(ypos==2))   // arriba
```

```
        {
          b1.pon();
          b2.ilumina(3,15);
          ypos=1;
        }
        if((car=='B')&&(ypos==1))  // abajo
        {
          b2.pon();
          b1.ilumina(3,15);
          ypos=2;
        }
    }
    if(raton.boton==1)
    {
        if((raton.x > 60)&&(raton.x<77)&&(raton.y>18)&&(raton.y<20))  // aceptar
        {
          b1.pon();
          b2.push(3,15);
          while(raton.boton==1)
            raton.activa();
          b2.pon();
          b1.pon();
          raton.desactiva();
          return(1);
        }
        if((raton.x > 60)&&(raton.x<77)&&(raton.y>21)&&(raton.y<23))  // ayuda
        {
          b2.pon();
          b1.push(3,15);
          while(raton.boton==1)
            raton.activa();
          b1.pon();
          b2.pon();
          raton.desactiva();
       p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
          salva_pant(0,0,79,24,(char *) p);
       a1.pon_marco();
       a1.control(0);
       restaura_pant(0,0,79,24,(char *) p);
       free(p);
       b1.ilumina(3,15);
       raton.inicia();
       ypos=2;
        }
    }
  } while(continua=='1');
  return(0);
}


int graficapor(int valorper[6])
{
  unsigned char *p;
  pantemerge a1(15,4,65,21,15,7,0,8,"ayuper.txt");
  mouse raton;
```

```
  char continua, car;
  div_t x;
  register int i, j, por, cuadros, ypos;
  pantalla c1(0,0,80,25,5,5,5,5,"grafiper.txt");
  botton e2(61,18,76,19,1,3,0,5,"ACEPTAR");
  botton e1(61,21,76,22,1,3,0,5," AYUDA");
  c1.pon_marco();
  e2.pon();
  e2.ilumina(3,15);
  e1.pon();
  c1.pon_texto(1,1,1,17,15);
  gotoxy(22,18); cprintf("  10 20 30 40 50 60 70 80 90 100     %");
  gotoxy(31,1); cprintf("TIPOS DE PERSONALIDAD");
/*  gotoxy(5,4); cprintf("REALISTA");
  gotoxy(5,6); cprintf("INVESTIGATIVO");
  gotoxy(5,8); cprintf("ARTISTICO");
  gotoxy(5,10); cprintf("SOCIAL");
  c1.escribe(5,12,15,5,"EMPRENDEDOR");
  c1.escribe(5,14,15,5,"CONVENCIONAL");*/
//  gotoxy(22,18);  cprintf("  10 20 30 40 50 60 70 80 90 100     %");
// c1.escribe(22,18,15,5,"  10 20 30 40 50 60 70 80 90 100     %");
  c1.escribe(5,21,14,5,"En  esta grafica comparamos los  resultados");
  c1.escribe(5,22,14,5,"obtenidos de tus intereses, habilidades y valores");
  c1.escribe(5,23,14,5,"con cada uno de los tipos de personalidad.");
  c1.lhor(23,17,61,15,0);
  c1.lver(23,3,16,15,0);
  _setcursortype(_NOCURSOR);
  gotoxy(23,17);  putch('À');
  for(i=0;i<6;i++)
  {
    por=valorper[i];
    x=div(por*30,100);
    cuadros=x.quot;
    for(j=0;j<cuadros;j++)
    {
        c1.escribe(24+j,4+i*2,9,5,"Ú");
        if(j!=0)
          c1.escribe(24+j,4+i*2+1,0,5,"ß");
        if(j==cuadros-1)
        {
          c1.escribe(24+j+1,4+i*2,0,5,"Ú");
          c1.escribe(24+j+1,4+i*2+1,0,5,"ß");
        }
    }
  }
  continua='1';
  raton.inicia();
  ypos=1;
  do
  {
    raton.activa();
    if(kbhit())
    {
        car=leetecla();
        if(car=='E')  // enter
        {
```

```
        if(ypos==1)                // aceptar
        {
       e1.pon();
          e2.push(3,15);
          delay(100);
          e2.pon();
          raton.desactiva();
          return(1);
        }
        if(ypos==2)                // ayuda
        {
          e1.push(3,15);
          delay(100);
          e1.pon();
       raton.desactiva();
       p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
          salva_pant(0,0,79,24,(char *) p);
       a1.pon_marco();
       a1.control(0);
       restaura_pant(0,0,79,24,(char *) p);
       free(p);
       e1.ilumina(3,15);
       e2.pon();
       raton.inicia();
        }
      }
      if((car=='A')&&(ypos==2))   // arriba
      {
        e1.pon();
        e2.ilumina(3,15);
        ypos=1;
      }
      if((car=='B')&&(ypos==1))   // abajo
      {
        e2.pon();
        e1.ilumina(3,15);
        ypos=2;
      }
}
if(raton.boton==1)
{
      if((raton.x > 60)&&(raton.x<77)&&(raton.y>18)&&(raton.y<20))   // aceptar
      {
        e1.pon();
        e2.push(3,15);
        while(raton.boton==1)
          raton.activa();
        e2.pon();
        e1.pon();
        raton.desactiva();
        return(1);
      }
      if((raton.x > 60)&&(raton.x<77)&&(raton.y>21)&&(raton.y<23))   // ayuda
      {
        e2.pon();
        e1.push(3,15);
```

```
            while(raton.boton==1)
              raton.activa();
            e1.pon();
            e2.pon();
          raton.desactiva();
          p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
            salva_pant(0,0,79,24,(char *) p);
          a1.pon_marco();
          a1.control(0);
          restaura_pant(0,0,79,24,(char *) p);
          free(p);
          e1.ilumina(3,15);
          e2.pon();
          raton.inicia();
          ypos=2;
          }
      }
    } while(continua=='1');
    return(0);
}


int pon_mejorih(char mejorih[4])
{

    unsigned char *p;
    pantemerge a1(15,4,65,21,15,7,0,8,"ayuint.txt");
    register int i;
    char mejor[3][24];
    int xpos;
    mouse raton;
    char continua, car;
    FILE *fp;
    pantalla d1(2,1,77,23,15,5,3,1,"orden.txt");
    bolton e2(35,21,50,22,1,3,0,5,"ACEPTAR");
    bolton e1(55,21,70,22,1,3,0,5,"AYUDA");
    d1.pon_fondo(1,3);
    d1.pon_marco();
    d1.pon_texto(13,7,1,17,15);
    e2.pon();
    e1.pon();
    e2.ilumina(3,15);
    d1.escribe(27,4,14,5,"PERFIL INTERES-HABILIDAD");
    _setcursortype(_NOCURSOR);
    d1.lhor(26,5,52,1,1);
    d1.lver(52,4,4,1,1);
    for(i=0;i<3;i++)
      switch(mejorih[i])
      {
          case '0': strcpy(mejor[i],"ASISTENCIAL");  break;
          case '1': strcpy(mejor[i],"EJECUTIVO-PERSUASIVO");  break;
          case '2':;strcpy(mejor[i],"VERBAL");  break;
          case '3': strcpy(mejor[i],"ARTISTICO-PLASTICO"); break;
          case '4': strcpy(mejor[i],"MUSICAL");  break;
          case '5': strcpy(mejor[i],"ORGANIZACION");  break;
```

```
        case '6': strcpy(mejor[i],"CIENTIFICO");  break;
        case '7': strcpy(mejor[i],"CALCULO");  break;
        case '8': strcpy(mejor[i],"MECANICO-CONSTRUCTIVO");  break;
        case '9': strcpy(mejor[i],"ACTIVIDAD AL AIRE LIBRE");
    }
d1.escribe(25,15,14,5,mejor[0]);
d1.escribe(25,17,14,5,mejor[1]);
d1.escribe(25,19,14,5,mejor[2]);
continua='1';
raton.inicia();
xpos=1;
do
{
   raton.activa();
   if(kbhit())
   {
        car=leetecla();
        if(car=='E')  // enter
        {
          if(xpos==1)              // aceptar
          {
            e2.push(3,15);
            delay(100);
            e2.pon();
            raton.desactiva();
            return(1);
          }
          if(xpos==2)              // ayuda
          {
            e1.push(3,15);
            delay(100);
            e1.pon();
          raton.desactiva();
          p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
             salva_pant(0,0,79,24,(char *) p);
          a1.pon_marco();
          a1.control(0);
          restaura_pant(0,0,79,24,(char *) p);
          free(p);
          e1.ilumina(3,15);
          e2.pon();
          raton.inicia();
          }
        }
        if((car=='C')&&(xpos==1))  // derecha
        {
          e2.pon();
          e1.ilumina(3,15);
          xpos=2;
        }
        if((car=='D')&&(xpos==2))  // izquierda
        {
          e1.pon();
          e2.ilumina(3,15);
          xpos=1;
        }
```

```
    }
    if(raton.boton==1)
    {
         if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))   // aceptar
         {
        e1.pon();
        e2.ilumina(3,15);
           e2.push(3,15);
           while(raton.boton==1)
             raton.activa();
           e2.pon();
           raton.desactiva();
           return(1);
         }
         if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))   // ayuda
         {
           e2.pon();
           e1.push(3,15);
           while(raton.boton==1)
             raton.activa();
           e1.pon();
           e2.pon();
           raton.desactiva();
        p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
           salva_pant(0,0,79,24,(char *) p);
        a1.pon_marco();
        a1.control(0);
        restaura_pant(0,0,79,24,(char *) p);
        free(p);
        e1.ilumina(3,15);
        raton.inicia();
        xpos=2;
         }
    }
  } while(continua=='1');
  return(1);
}




int main_resulta()
{
  while(!(introduccres("res1.txt")));
    while(!(introduccres("res2.txt")))
         while(!(introduccres("res1.txt")));
  return(1);
}
```

```
// ARCHIVO FINAL.CPP.- Es  parte  de  la  interface  de usuario y se
// encarga de deplegar los   resultados finales y de dar la explicacion
// de los mismos.


// Realiza la £ltima labor de la interface de usuario:
// la exhibici¢n de los resultados y la explicaci¢n de
// los mismos.

#include "c:\tesis\marco.h"

int introducc1final(char archi[11]);  // despliega la introduccion final
int presentafin(char mejorper[4]);    // presenta un menu con las 5 mejores carreras
int muestra(int nocar, char mejorper[4]);  // muestra el perfil de una carrera
int presentatodas(char mejorper[4]);  // muestra la lista de todas las carreras
int pantallita();  // despliega un apantalla de ayuda al perfil de una carrera


// Esta funcion esta definida en este archivo aunque no se ejecuta desde este
//int mainfinal(char mejorper[4]);
// coordina esta parte de la interface de usuario



int introducc1final(char archi[11])
{
  int xpos;
  char car;
  char continua;
  mouse raton;
  _setcursortype(_NOCURSOR);
  pantalla a1(2,1,77,23,15,5,3,1,archi);
  botton b2(35,21,50,22,1,3,0,5,"ADELANTE");
  botton b1(55,21,70,22,1,3,0,5,"P g.Ant.");
  a1.pon_fondo(1,3);
  a1.pon_marco();
  b2.pon();
  b2.ilumina(3,15);
  b1.pon();
  a1.escribe(30,4,14,5,"RESULTADOS FINALES");
  a1.lhor(23,5,55,1,1);
  a1.lver(55,4,4,1,1);
  continua='1';
  raton.inicia();
  xpos=1;
   if((strcmpi(archi,"final1.txt")))
     a1.pon_texto(9,9,1,17,15);
   else
      a1.pon_texto(9,7,1,17,15);
  do
  {
     raton.activa();
     If(kbhit())
     {
          car=leetecla();
```

```
        if(car=='E')  // enter
        {
          if(xpos==1)
          {
            b2.push(3,15);
            delay(100);
            b2.pon();
            raton.desactiva();
            return(1);
          }
          if(xpos==2)
          {
            b1.push(3,15);
            delay(100);
            b1.pon();
            raton.desactiva();
            return(0);
          }
        }
        if((car=='C')&&(xpos==1))  // derecha
        {
          b2.pon();
          b1.ilumina(3,15);
          xpos=2;
        }
        if((car=='D')&&(xpos==2))  // izquierda
        {
          b1.pon();
          b2.ilumina(3,15);
          xpos=1;
        }
    }
    if(raton.boton==1)
    {
        if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))  // aceptar
        {
          b2.push(3,15);
          while(raton.boton==1)
            raton.activa();
          b2.pon();
          raton.desactiva();
          return(1);
        }
        if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // cancelar
        {
          b2.pon();
          b1.push(3,15);
          while(raton.boton==1)
            raton.activa();
          b1.pon();
          raton.desactiva();
          return(0);
        }
    }
  } while(continua=='1');
  return(0);
```

```
}


int presentafin(char mejorper[4])
{
  typedef struct
  {
    char nombre[43];
    int coefi;
    int nocar;
  } carreras;
  carreras carrera[5];
  FILE *fp;
  unsigned char *p;
  register int i,j;
  int xpos;
  char car;
  char continua;
  mouse raton;
  _setcursortype(_NOCURSOR);
  pantalla a1(2,1,77,23,15,5,3,1,"      ");
  botton b2(40,21,55,22,1,3,0,5,"TERMINAR");
  botton b1(59,21,74,22,1,3,0,5,"P g.Ant.");
  botton e1(9,6,14,7,1,3,0,5,"1");
  botton e2(9,9,14,10,1,3,0,5,"2");
  botton e3(9,12,14,13,1,3,0,5,"3");
  botton e4(9,15,14,16,1,3,0,5,"4");
  botton e5(9,18,14,19,1,3,0,5,"5");
  botton e6(9,21,14,22,1,3,0,5,"6");
  a1.pon_fondo(1,3);
  a1.pon_marco();
  b1.pon();
  b2.pon();
  e1.pon();
  e2.pon();
  e3.pon();
  e4.pon();
  e5.pon();
  e6.pon();
  e1.ilumina(3,15);
  a1.escribe(30,4,14,5,"RESULTADOS FINALES");
  a1.lhor(23,5,55,1,1);
  a1.lver(55,4,4,1,1);
  if((fp=fopen("memoria1.dat","r"))==NULL)
  {
    error(1,"MEMORIA.DAT");
    return(0);
  }
 for(i=0;i<5;i++)
 {
   carrera[i].nombre[42]='\0';
   for(j=0;j<42;j++)
     fscanf(fp,"%c",&carrera[i].nombre[j]);
   fscanf(fp," %d %d\n",&carrera[i].nocar,&carrera[i].coefi);
   a1.escribe(18,7+i*3,15,5,carrera[i].nombre);
```

```
}
a1.escribe(18,22,15,5,"LISTA DE CARRERAS");
continua='1';
a1.escribe(18,7,11,5,carrera[0].nombre);
raton.inicia();
xpos=1;
do
{
  raton.activa();
  if(kbhit())
  {
      car=leetecla();
      if(car=='E')   // enter
      {
      switch(xpos)
      {
        case 1:
              {
                e1.push(3,15);
                delay(100);
                e1.pon();
              e1.ilumina(3,15);
                raton.desactiva();
              p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de
video*/
                    salva_pant(0,0,79,24,(char *) p);
              muestra(carrera[0].nocar,mejorper);
                    restaura_pant(0,0,79,24,(char *) p);
                    free(p);
              raton.inicia();
               }
             break;
        case 2:
              {
                e2.push(3,15);
                delay(100);
                e2.pon();
              e2.ilumina(3,15);
                raton.desactiva();
              p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de
video*/
                    salva_pant(0,0,79,24,(char *) p);
              muestra(carrera[1].nocar,mejorper);
                    restaura_pant(0,0,79,24,(char *) p);
                    free(p);
              raton.inicia();
               }
             break;
        case 3:
              {
                e3.push(3,15);
                delay(100);
                e3.pon();
              e3.ilumina(3,15);
                raton.desactiva();
```

```
            p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de
video*/
                  salva_pant(0,0,79,24,(char *) p);
            muestra(carrera[2].nocar,mejorper);
                  restaura_pant(0,0,79,24,(char *) p);
                  free(p);
            raton.inicia();
              }
            break;
      case 4:
            {
              e4.push(3,15);
              delay(100);
              e4.pon();
            e4.ilumina(3,15);
              raton.desactiva();
            p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de
video*/
                  salva_pant(0,0,79,24,(char *) p);
            muestra(carrera[3].nocar,mejorper);
                  restaura_pant(0,0,79,24,(char *) p);
                  free(p);
            raton.inicia();
              }
            break;
      case 5:
            {
              e5.push(3,15);
              delay(100);
              e5.pon();
            e5.ilumina(3,15);
              raton.desactiva();
            p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de
video*/
                  salva_pant(0,0,79,24,(char *) p);
            muestra(carrera[4].nocar,mejorper);
                  restaura_pant(0,0,79,24,(char *) p);
                  free(p);
            raton.inicia();
              }
            break;
      case 6:
            {

              e6.push(3,15);
              delay(100);
              e6.pon();
            e6.ilumina(3,15);
              raton.desactiva();
            p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de
video*/
                  salva_pant(0,0,79,24,(char *) p);
            presentatodas(mejorper);
                  restaura_pant(0,0,79,24,(char *) p);
                  free(p);
            raton.inicia();
```

```
                }
              break;
        case 7:
                {
                  b2.push(3,15);
                  delay(100);
                  b2.pon();
                  raton.desactiva();
                  return(1);
                }
              break;
        case 8:
                {
                  b1.push(3,15);
                  delay(100);
                  b1.pon();
                  raton.desactiva();
                  return(0);
                }
              break;
      }
      }
       if(car=='C')  // derecha
       {
      if(xpos==6)
      {
            e6.pon();
        a1.escribe(18,7+(5)*3,15,5,"LISTA DE CARRERAS");
            b2.ilumina(3,15);
            xpos=7;
      }
      else
      if(xpos==7)
      {
        b2.pon();
        b1.ilumina(3,15);
        xpos=8;
      }
    }
       if(car=='D')  // Izquierda
       {
      if(xpos==8)
      {
        b1.pon();
        b2.ilumina(3,15);
        xpos--;
      }
      else
      if(xpos==7)
      {
            b2.pon();
            e6.ilumina(3,15);
        a1.escribe(18,7+(5)*3,11,5,"LISTA DE CARRERAS");
            xpos=6;
      }
    }
```

```
if(car=='A') // arriba
{
  switch(xpos)
  {
    case(2):
          {
           e2.pon();
           e1.ilumina(3,15);
           xpos--;
          }
          break;
    case(3):
          {
           e3.pon();
           e2.ilumina(3,15);
           xpos--;
          }
          break;
    case(4):
          {
           e4.pon();
           e3.ilumina(3,15);
           xpos--;
          }
          break;
    case(5):
          {
           e5.pon();
           e4.ilumina(3,15);
           xpos--;
          }
          break;
    case(6):
          {
           e6.pon();
           e5.ilumina(3,15);
           xpos--;
          }
          break;
  }
  if((xpos!=5)&&(xpos!=7)&&(xpos!=8))
  {
    a1.escribe(18,7+(xpos)*3,15,5,carrera[xpos].nombre);
    a1.escribe(18,7+(xpos-1)*3,11,5,carrera[xpos-1].nombre);
  }
  else
  if(xpos==5)
  {
    a1.escribe(18,7+(5)*3,15,5,"LISTA DE CARRERAS");
    a1.escribe(18,7+(xpos-1)*3,11,5,carrera[4].nombre);
  }
}
if(car=='B')  // abajo
{
  switch(xpos)
  {
```

```
        case(1):
              {
               e1.pon();
               e2.ilumina(3,15);
               xpos++;
              }
              break;
        case(2):
              {
               e2.pon();
               e3.ilumina(3,15);
               xpos++;
              }
              break;
        case(3):
              {
               e3.pon();
               e4.ilumina(3,15);
               xpos++;
              }
              break;
        case(4):
              {
               e4.pon();
               e5.ilumina(3,15);
               xpos++;
              }
              break;
        case(5):
              {
               e5.pon();
               e6.ilumina(3,15);
               xpos++;
              }
              break;
      }
      if((xpos!=6)&&(xpos!=7)&&(xpos!=8))
      {
        a1.escribe(18,7+(xpos-2)*3,15,5,carrera[xpos-2].nombre);
        a1.escribe(18,7+(xpos-1)*3,11,5,carrera[xpos-1].nombre);
      }
      else
      if(xpos==6)
      {
        a1.escribe(18,7+(xpos-1)*3,11,5,"LISTA DE CARRERAS");
        a1.escribe(18,7+(xpos-2)*3,15,5,carrera[4].nombre);
      }
    }
  }
  if(raton.boton==1)
  {
      if((raton.x > 39)&&(raton.x<56)&&(raton.y>21)&&(raton.y<23))   // aceptar
      {
        b2.push(3,15);
        while(raton.boton==1)
          raton.activa();
```

```
    b2.pon();
    raton.desactiva();
    return(1);
  }
  if((raton.x > 58)&&(raton.x<75)&&(raton.y>21)&&(raton.y<23))  // p g. ant.
  {
    b2.pon();
  b1.ilumina(3,15);
    b1.push(3,15);
    while(raton.boton==1)
      raton.activa();
    b1.pon();
    raton.desactiva();
    return(0);
  }
if((raton.x > 8)&&(raton.x<15)&&(raton.y>6)&&(raton.y<8)) // carrera 0
  {
    b2.pon(); b1.pon();e2.pon(); e3.pon(); e4.pon(); e5.pon(); e6.pon();
  e1.ilumina(3,15);
  xpos=1;
  for(i=0;i<5;i++)
    a1.escribe(18,7+(i)*3,15,5,carrera[i].nombre);
  a1.escribe(18,22,15,5,"LISTA DE CARRERAS");
  a1.escribe(18,7+(xpos-1)*3,11,5,carrera[xpos-1].nombre);
    e1.push(3,15);
    while(raton.boton==1)
      raton.activa();
    raton.desactiva();
  e1.pon();
  e1.ilumina(3,15);
  p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
    salva_pant(0,0,79,24,(char *) p);
  muestra(carrera[0].nocar,mejorper);
    restaura_pant(0,0,79,24,(char *) p);
    free(p);
  raton.inicia();
  }
if((raton.x > 8)&&(raton.x<15)&&(raton.y>9)&&(raton.y<11)) //carrera 2
  {
    b2.pon(); b1.pon();e1.pon(); e3.pon(); e4.pon(); e5.pon(); e6.pon();
  e2.ilumina(3,15);
  xpos=2;
  for(i=0;i<5;i++)
    a1.escribe(18,7+(i)*3,15,5,carrera[i].nombre);
  a1.escribe(18,22,15,5,"LISTA DE CARRERAS");
  a1.escribe(18,7+(xpos-1)*3,11,5,carrera[xpos-1].nombre);
    e2.push(3,15);
    while(raton.boton==1)
      raton.activa();
    raton.desactiva();
  e2.pon();
  e2.ilumina(3,15);
  p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
    salva_pant(0,0,79,24,(char *) p);
  muestra(carrera[1].nocar,mejorper);
    restaura_pant(0,0,79,24,(char *) p);
```

```
      free(p);
    raton.inicia();
    }
  if((raton.x > 8)&&(raton.x<15)&&(raton.y>12)&&(raton.y<14))//carrera 3
    {
      b2.pon(); b1.pon();e1.pon(); e2.pon(); e4.pon(); e5.pon(); e6.pon();
    e3.ilumina(3,15);
    xpos=3;
    for(i=0;i<5;i++)
      a1.escribe(18,7+(i)*3,15,5,carrera[i].nombre);
    a1.escribe(18,22,15,5,"LISTA DE CARRERAS");
    a1.escribe(18,7+(xpos-1)*3,11,5,carrera[xpos-1].nombre);
      e3.push(3,15);
      while(raton.boton==1)
        raton.activa();
      raton.desactiva();
    e3.pon();
    e3.ilumina(3,15);
    p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
      salva_pant(0,0,79,24,(char *) p);
    muestra(carrera[2].nocar,mejorper);
      restaura_pant(0,0,79,24,(char *) p);
      free(p);
    raton.inicia();
    }
  if((raton.x > 8)&&(raton.x<15)&&(raton.y>15)&&(raton.y<17))//carrera 4
    {
      b2.pon(); b1.pon();e1.pon(); e2.pon(); e3.pon(); e5.pon(); e6.pon();
    e4.ilumina(3,15);
    xpos=4;
    for(i=0;i<5;i++)
      a1.escribe(18,7+(i)*3,15,5,carrera[i].nombre);
    a1.escribe(18,22,15,5,"LISTA DE CARRERAS");
    a1.escribe(18,7+(xpos-1)*3,11,5,carrera[xpos-1].nombre);
      e4.push(3,15);
      while(raton.boton==1)
        raton.activa();
      raton.desactiva();
    e4.pon();
    e4.ilumina(3,15);
    p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
      salva_pant(0,0,79,24,(char *) p);
    muestra(carrera[3].nocar,mejorper);
      restaura_pant(0,0,79,24,(char *) p);
      free(p);
    raton.inicia();
    }
  if((raton.x > 8)&&(raton.x<15)&&(raton.y>18)&&(raton.y<20))//carrera 5
    {
      b2.pon(); b1.pon();e1.pon(); e2.pon(); e3.pon(); e4.pon(); e6.pon();
    e5.ilumina(3,15);
    xpos=5;
    for(i=0;i<5;i++)
      a1.escribe(18,7+(i)*3,15,5,carrera[i].nombre);
    a1.escribe(18,22,15,5,"LISTA DE CARRERAS");
    a1.escribe(18,7+(xpos-1)*3,11,5,carrera[xpos-1].nombre);
```

```
            e5.push(3,15);
            while(raton.boton==1)
              raton.activa();
            raton.desactiva();
          e5.pon();
          e5.ilumina(3,15);
          p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
            salva_pant(0,0,79,24,(char *) p);
          muestra(carrera[4].nocar,mejorper);
            restaura_pant(0,0,79,24,(char *) p);
            free(p);
          raton.inicia();
          }
        if((raton.x > 8)&&(raton.x<15)&&(raton.y>21)&&(raton.y<23))//lista de carreras
          {
            b2.pon(); b1.pon();e1.pon(); e2.pon(); e3.pon(); e4.pon(); e5.pon();
          e6.ilumina(3,15);
          xpos=6;
          for(i=0;i<5;i++)
            a1.escribe(18,7+(i)*3,15,5,carrera[i].nombre);
          a1.escribe(18,22,11,5,"LISTA DE CARRERAS");
            e6.push(3,15);
            while(raton.boton==1)
              raton.activa();
            raton.desactiva();
          e6.pon();
          e6.ilumina(3,15);
          p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
            salva_pant(0,0,79,24,(char *) p);
          presentatodas(mejorper);
            restaura_pant(0,0,79,24,(char *) p);
            free(p);
          raton.inicia();
          }
      }
    } while(continua=='1');
    fclose(fp);
    return(0);
}



int pantallita()
{
    char continua,car;
    mouse raton;
    pantalla a1(13,6,63,17,15,7,0,8,"      ");
    botton b1(40,15,55,16,1,3,0,7,"ACEPTAR");
    _setcursortype(_NOCURSOR);
    a1.pon_marco();
    b1.pon();
    b1.ilumina(3,15);
    a1.escribe(20,9,0,7,"A continuación se muestra el perfil");
    a1.escribe(20,10,0,7,"de la carrera, los aspectos que");
    a1.escribe(20,11,0,7,"aparecen marcados con ¯, son");
```

```
  a1.escribe(20,12,0,7,"aquellos que coinciden con tu perfil.");
  continua='1';
  raton.inicia();
  do
  {
    raton.activa();
    if(kbhit())
    {
          car=leetecla();
          if(car=='E')  // enter
          {
         raton.desactiva();
            b1.push(3,15);
            delay(100);
            b1.pon();
         b1.ilumina(3,15);
         continua='0';
      }
    }
    if(raton.boton==1)
          if((raton.x > 39)&&(raton.x<56)&&(raton.y>15)&&(raton.y<17))  // aceptar
          {
            b1.push(3,15);
            while(raton.boton==1)
              raton.activa();
            b1.pon();
         b1.ilumina(3,15);
            raton.desactiva();
         continua='0';
          }
  } while(continua=='1');
}


int muestra(int nocar, char mejorper[4])
{
  FILE *fp, *fc, *fd;
  unsigned char *p;
  register int i,j,k;
  int xpos, reng, tot, oc, ar;
  char car, ocupa[59],area[41],per[4], nomocupa[70], nomarea[35];
  char continua, nombre[43];
  mouse raton;
  pantalla a1(2,1,77,23,15,5,3,1,"      ");
  botton b1(55,21,70,22,1,3,0,5,"ACEPTAR");
  _setcursortype(_NOCURSOR);
  a1.pon_fondo(1,3);
  a1.pon_marco();
  b1.pon();
  if((fp=fopen("memoria.dat","r"))==NULL)
  {
    error(1,"MEMORIA.DAT");
    return(0);
  }
  // saltar los tres primeros renglones
```

```
for(i=0;i<3;i++)
  do
       fscanf(fp,"%c",&car);
  while(car!='\n');
// leer ocupacion
for(i=0;i<59;i++)
  fscanf(fp,"%c",&ocupa[i]);
fscanf(fp,"%c",&car);
// leer areas de memoria de trabajo
for(i=0;i<41;i++)
  fscanf(fp,"%c",&area[i]);
fscanf(fp,"\n");
fclose(fp);
// abrir el archivo carreas.dat
if((fp=fopen("carreras.dat","r"))==NULL)
{
  error(1,"CARRERAS.DAT");
  return(0);
}
if((fc=fopen("ocupa1.txt","r"))==NULL)
{
  error(1,"OCUPA1.TXT");
  return(0);
}
if((fd=fopen("area1.txt","r"))==NULL)
{
  error(1,"AREA1.TXT");
  return(0);
}
reng=4;   // indica el valor en y para comenzar a escribir
//saltar los primeros dos renglones
for(i=0;i<2+nocar;i++)
  do
       fscanf(fp,"%c",&car);
  while(car!='\n');
// leer nombre de la carrera
for(j=0;j<42;j++)
  fscanf(fp,"%c",&nombre[j]);
nombre[42]='\0';
a1.escribe(25,3,14,5,nombre);
a1.escribe(7,reng++,11,5,"Personalidad");
fscanf(fp,"%c%c%c",&per[0],&per[1],&per[2]);
per[3]='\0';
for(i=0;i<3;i++)
{
  if(per[i]!=' ')
  {
    switch(per[i])
    {
      case 'R': a1.escribe(7,reng,15,5,"[ ] Realista");
           break;
      case 'I': a1.escribe(7,reng,15,5,"[ ] Investigativo");
           break;
      case 'A':a1.escribe(7,reng,15,5,"[ ] Artistico");
           break;
      case 'S':a1.escribe(7,reng,15,5,"[ ] Social");
```

```
            break;
        case 'E':a1.escribe(7,reng,15,5,"[ ] Emprendedor");
            break;
        case 'C':a1.escribe(7,reng,15,5,"[ ] Convencional");
            break;
      }
      if((per[i]==mejorper[0])||(per[i]==mejorper[1])||(per[i]==mejorper[2]))
          a1.escribe(8,reng,11,5,"*");
      reng++;
    }
  }
  reng++;
  // leer rama y campo desde la columna 47
  a1.escribe(7,reng++,11,5,"Rama y campo ocupacional");
  fscanf(fp," %d ",&tot);
  for(i=0;i<tot;i++)
  {
      fscanf(fp,"%d ",&oc);
      // Leer el nombre de la ocupacion
      fseek(fc,0,0);
      for(j=0;j<oc;j++)   // saltar renglones
      do
        fscanf(fc,"%c",&car);
      while(car!='\n');
      k=0;
      do
      {
        fscanf(fc,"%c",&car);
        if(car!='\n')
          nomocupa[k++]=car;
      }while(car!='\n');
      for(j=k;j<69;j++)
        nomocupa[j]=' ';
      nomocupa[69]='\0';
      a1.escribe(7,reng++,15,5,nomocupa);
      if(ocupa[oc]=='p')
        a1.escribe(8,reng-1,11,5,"*");
  }
  reng++;
  // leer area
  a1.escribe(7,reng++,11,5,"Areas de estudio profesional");
  fscanf(fp,"%d ",&tot);
  for(i=0;i<tot;i++)
  {
    fscanf(fp,"%d%c",&ar,&car);
    fseek(fd,0,0);
    for(k=0;k<ar-1;k++)   // saltar renglones
      do
        fscanf(fd,"%c",&car);
      while(car!='\n');
    k=0;
    do
    {
      fscanf(fd,"%c",&car);
      if(car!='\n')
        nomarea[k++]=car;
```

```
    }while(car!='\n');
    for(j=k;j<34;j++)
      nomarea[j]=' ';
    nomarea[34]='\0';
    a1.escribe(7,reng++,15,5,nomarea);
    if(area[ar-1]=='p')   // -1 porque el arreglo comienza en 0 y las claves en 1
      a1.escribe(8,reng-1,11,5," ");
  }
  fclose(fp);
  fclose(fc);
  fclose(fd);
  // despliega mensaje de ayuda
  p = (unsigned char *) malloc((160) * (25));  //asignar memoria para el buffer de video
  salva_pant(0,0,79,24,(char *) p);
  pantallita();
  restaura_pant(0,0,79,24,(char *) p);
  free(p);
  b1.ilumina(3,15);
  continua='1';
  raton.inicia();
  do
  {
    raton.activa();
    if(kbhit())
    {
      car=leetecla();
        if(car=='E')  // enter
        {
          b1.push(3,15);
          delay(100);
          b1.pon();
          raton.desactiva();
        return(1);
      }
    }
    if(raton.boton==1)
        if(((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))   // aceptar
        {
          b1.push(3,15);
          while(raton.boton==1)
            raton.activa();
          b1.pon();
          raton.desactiva();
          return(1);
        }
  } while(continua=='1');
  return(1);
}




int presentatodas(char mejorper[4])
{
  typedef struct
```

```
        raton.desactiva();
      b2.pon();                                  // CARRERA
        b1.push(3,15);
      delay(100);
      b1.pon();
      b1.ilumina(3,15);
      p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
        salva_pant(0,0,79,24,(char *) p);
      //pantallita();
      muestra(carrera[actual].nocar,mejorper);
        restaura_pant(0,0,79,24,(char *) p);
        free(p);
      raton.inicia();
    }
    else
    {
      raton.desactiva();
      b1.pon();
      b2.push(3,15);
      delay(100);
      b2.pon();
      continua='0';
    }
    }
  if(car=='Q')  // pagina abajo
  {
    if(yfin!=448)
    {
      yini+=14;
      yfin+=14;
      if(yfin>448)
      {
        yfin=448;
        yini=448-14;
      }
      // saltar renglones
      fseek(fp,0,0);
      for(i=0;i<yini;i++)
      do
        fscanf(fp,"%c",&car);
      while(car!='\n');
      for(i=yini;i<=yfin;i++)
      {
        for(j=0;j<42;j++)
          fscanf(fp,"%c",&carrera[i-yini].nombre[j]);
        fscanf(fp," %d %d\n",&carrera[i-yini].nocar,&carrera[i-yini].coefi);
        a1.escribe(10,6+i-yini,15,5,carrera[i-yini].nombre);
        cprintf("    %3d/100",carrera[i-yini].coefi);
      }
    }
    a1.escribe(10,6+actual,11,5,carrera[actual].nombre);
    cprintf("    %3d/100",carrera[actual].coefi);
  }
  if(car=='I')  // pagina arriba
  {
    if(yfin!=0)
```

```
        for(i=0;i<yini;i++)
        do
          fscanf(fp,"%c",&car);
        while(car!='\n');
        for(i=yini;i<=yfin;i++)
        {
          for(j=0;j<42;j++)
          fscanf(fp,"%c",&carrera[i-yini].nombre[j]);
          fscanf(fp," %d %d\n",&carrera[i-yini].nocar,&carrera[i-yini].coefi);
          a1.escribe(10,6+i-yini,15,5,carrera[i-yini].nombre);
          cprintf("    %3d/100",carrera[i-yini].coefi);
        }
      }
    }
    else
      --actual;
    a1.escribe(10,6+actual,11,5,carrera[actual].nombre);
    cprintf("    %3d/100",carrera[actual].coefi);
    a1.escribe(10,6+actual+1,15,5,carrera[actual+1].nombre);
    cprintf("    %3d/100",carrera[actual+1].coefi);

  }
  if(car=='B')  // abajo
  {
    if(actual==14)
    {
      if(yfin!=448)
      {
        ++yini;
        ++yfin;
        // saltar renglones
        fseek(fp,0,0);
        for(i=0;i<yini;i++)
        do
          fscanf(fp,"%c",&car);
        while(car!='\n');
        for(i=yini;i<=yfin;i++)
        {
          for(j=0;j<42;j++)
            fscanf(fp,"%c",&carrera[i-yini].nombre[j]);
          fscanf(fp," %d %d\n",&carrera[i-yini].nocar,&carrera[i-yini].coefi);
          a1.escribe(10,6+i-yini,15,5,carrera[i-yini].nombre);
          cprintf("    %3d/100",carrera[i-yini].coefi);
        }
      }
    }
    else
      actual++;
    a1.escribe(10,6+actual,11,5,carrera[actual].nombre);
    cprintf("    %3d/100",carrera[actual].coefi);
    a1.escribe(10,6+actual-1,15,5,carrera[actual-1].nombre);
    cprintf("    %3d/100",carrera[actual-1].coefi);

  }
}
if(raton.boton==1)
```

```
    {
        if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))   // MOSTRAR
        {
        b2.pon();                                          // CARRERA
          b1.push(3,15);
          while(raton.boton==1)
            raton.activa();
          raton.desactiva();
        b1.pon();
        b1.ilumina(3,15);
        p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
          salva_pant(0,0,79,24,(char *) p);
        //pantallita();
        muestra(carrera[actual].nocar,mejorper);
          restaura_pant(0,0,79,24,(char *) p);
          free(p);
        raton.inicia();
      }
        if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))   // TERMINAR
        {
          b1.pon();
        b2.ilumina(3,15);
          b2.push(3,15);
          while(raton.boton==1)
            raton.activa();
          raton.desactiva();
        b2.pon();
        b2.ilumina(3,15);
        b1.pon();
        continua='0';
          //return(0);
        }
    }
  } while(continua=='1');
  fclose(fp);
  return(0);
}



int mainfinal(char mejorper[4])
{
  while(!(introducc1final("final1.txt")));
  while(!(introducc1final("final2.txt")))
    while(!(introducc1final("final1.txt")));
  while(!presentafin(mejorper))
    while(!(introducc1final("final2.txt")))
      while(!(introducc1final("final1.txt")));
  return(0);
}
```

```
// ARCHIVO IMPRIME.CPP.- Es parte de la interface con el usuario y
// se encarga de imprimir el reporte de resultados.


#include <c:\tesis\marco.h>

// Modulo para la impresion del reporte

void inicialmp();  // inicializa la impresora
int chkprn();  /* regresa 1 si hay error en prn */


void inicialmp()
{
  biosprint(1,' ',0);
  fprintf(stdprn,"\n\n\n\n\n");
}



int chkprn()  /* regresa 1 si hay error en prn */
{
  int status, abyte=0;
  status = biosprint(2, abyte, 0);
  if (status & 0x20)  {
    error(2,"          ");          /* "Out of paper"2*/
    return(1);
  }
  else
    if(status & 0x08)  {
        error(3,"     ");        /* "I/O error" */
        return(1);
    }
    else
        if (status & 0x01)  {
          error(3,"        ");      /*"Device time out*/
          return(1);
        }
    else
        if (!(status & 0x10))  {
          error(4,"     ");          /* not selected */
          return(1);
        }
        else  {
            inicialmp();
          return(0);
        }
}



int imprime(int valorper[6])
{
```

```
FILE *fc;
register int i,x;
int evinter[10], evhabi[10];
char c;
char nombre[41], fecha[9], edad[3], sexo[11];
char estado[11], escolaridad[21], ocupa[41];
char domicilio[58], telefono[11];
pantalla a1(13,6,63,17,15,7,0,8,"      ");
unsigned char *p;
p = (unsigned char *) malloc((160) * (25)); //asignar memoria para el buffer de video
salva_pant(0,0,79,24,(char *) p);
_setcursortype(_NOCURSOR);
a1.pon_marco();
a1.escribe(20,9,0,7,"        IMPRIMIENDO RESULTADOS");
a1.escribe(20,13,15,7,"ÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛÛ");
x=0;
if(chkprn())
{
  restaura_pant(0,0,79,24,(char *) p);
  free(p);
  return(1);
}

if((fc=fopen("datper.dat","r"))==NULL)
{
  error(1,"DATPER.DAT");
  return(0);
}
// leer los datos personales
fscanf(fc,"%[A-Za-z 0-9.,:-]\n%[A-Za-z 0-9.,;-]\n%[A-Za-z 0-9.,;-]\n%[A-Za-z 0-9.,;-]\n%[A-Za-z 0-9.,;-]\n",nombre,fecha,edad,sexo,estado);
fscanf(fc,"%[A-Za-z 0-9.,:-]\n%[A-Za-z 0-9.,;-]\n%[A-Za-z 0-9.,#;-]\n%[A-Za-z 0-9.,;-]\n",escolaridad, ocupa, domicilio, telefono);
fclose(fc);
// leer valor de intereses
if((fc=fopen("memoria.dat","r"))==NULL)
 {
  error(1,"MEMORIA.DAT");
  return(0);
}
for(i=0;i<10;i++)
  fscanf(fc,"%d ",&evinter[i]);
fscanf(fc,"\n");
// leer valor de habilidades
for(i=0;i<10;i++)
  fscanf(fc,"%d ",&evhabi[i]);
fscanf(fc,"\n");
fclose(fc);
a1.escribe(20+x++,13,9,7,"Û");
fprintf(stdprn,"                    ***********\n\r");
a1.escribe(20+x++,13,9,7,"Û");
fprintf(stdprn,"                    * RESUMEN *\n\r");
a1.escribe(20+x++,13,9,7,"Û");
fprintf(stdprn,"                    ***********\n\n\n\n\n\r");
a1.escribe(20+x++,13,9,7,"Û");
fprintf(stdprn,"   -------------------- DATOS PERSONALES --------------------\n\n\r");
```

```
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"              NOMBRE: %s\n\r",nombre);
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"  FECHA DE NACIMIENTO: %s\n\r",fecha);
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"               EDAD: %s\n\r",edad);
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"               SEXO: %s\n\r",sexo);
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"       ESTADO CIVIL: %s\n\r",estado);
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"       ESCOLARIDAD: %s\n\r",escolaridad);
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"          OCUPACION: %s\n\r",ocupa);
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"          DOMICILIO: %s\n\r",domicilio);
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"           TELEFONO: %s\n\n\n\n\n\n\r",telefono);
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"  ------------------ INTERESES Y HABILIDADES ------------------\n\n\r");
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"                          INTERES   HABILIDAD\n\r");
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"       ASISTENCIAL:   %3d%c
%3d%c\n\r",evinter[0]*100/24,'%',evhabi[0]*100/24,'%');
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"  EJECUTIVO-PERSUASIVO:   %3d%c
%3d%c\n\r",evinter[1]*100/24,'%',evhabi[1]*100/24,'%');
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"          VERBAL:   %3d%c
%3d%c\n\r",evinter[2]*100/24,'%',evhabi[2]*100/24,'%');
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"   ARTISTICO-PLASTICO:   %3d%c
%3d%c\n\r",evinter[3]*100/24,'%',evhabi[3]*100/24,'%');
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"          MUSICAL:   %3d%c
%3d%c\n\r",evinter[4]*100/24,'%',evhabi[4]*100/24,'%');
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"     ORGANIZACION:   %3d%c
%3d%c\n\r",evinter[5]*100/24,'%',evhabi[5]*100/24,'%');
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"       CIENTIFICO:   %3d%c
%3d%c\n\r",evinter[6]*100/24,'%',evhabi[6]*100/24,'%');
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"          CALCULO:   %3d%c
%3d%c\n\r",evinter[7]*100/24,'%',evhabi[7]*100/24,'%');
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"  MECANICO-CONSTRUCT.:   %3d%c
%3d%c\n\r",evinter[8]*100/24,'%',evhabi[8]*100/24,'%');
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"   ACT. AL AIRE LIBRE:   %3d%c
%3d%c\n\n\n\n\n\r",evinter[9]*100/24,'%',evhabi[9]*100/24,'%');
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"  ---------------------- PERSONALIDAD ----------------------\n\n\r");
a1.escribe(20+x++,13,9,7,"Ú");
fprintf(stdprn,"          REALISTA: %d%c\n\r",valorper[0],'%');
```

```
  a1.escribe(20+x++,13,9,7,"Ú");
  fprintf(stdprn,"          INVESTIGATIVO: %d%c\n\r",valorper[1],'%');
  a1.escribe(20+x++,13,9,7,"Ú");
  fprintf(stdprn,"             ARTISTICO: %d%c\n\r",valorper[2],'%');
  a1.escribe(20+x++,13,9,7,"Ú");
  fprintf(stdprn,"                SOCIAL: %d%c\n\r",valorper[3],'%');
  a1.escribe(20+x++,13,9,7,"Ú");
  fprintf(stdprn,"          EMPRENDEDOR: %d%c\n\r",valorper[4],'%');
  a1.escribe(20+x++,13,9,7,"Ú");
  fprintf(stdprn,"          CONVENCIONAL: %d%c\n\r",valorper[5],'%');
  a1.escribe(20+x++,13,9,7,"Ú");
  fprintf(stdprn,"%c","\xC');
  a1.escribe(20+x++,13,9,7,"Ú");
  a1.escribe(20+x++,13,9,7,"Ú");
  a1.escribe(20+x++,13,9,7,"Ú");
  a1.escribe(20+x++,13,9,7,"Ú");
  a1.escribe(20+x++,13,9,7,"Ú");
  restaura_pant(0,0,79,24,(char *) p);
  free(p);
  return(1);
}
```

// ARCHIVO GUIA.CPP.- Coordina la seccion de la guia de carreras.

```
#include "c:\tesis\marco.h"

int presenta(char archivo[13], int ini, int fin);
   // pone en pantalla el contenido de un archivo permitiendo desplazarse
int presentadir(char archivo[13], int ini, int fin);
   // muestra el directorio de instituciones
int menu_por_carrera(char archi[13], int tam);
   // muestra las carreras contenidas en la guia



int presenta(char archivo[13], int ini, int fin)
{
  register int xpos, inicio;
  int color;
  char continua, car;
  mouse raton;
  _setcursortype(_NOCURSOR);
  pantalla a1(2,1,77,23,15,5,3,1,archivo);
  botton b1(55,21,70,22,1,3,0,5," ACEPTAR");
  a1.pon_fondo(1,3);
  a1.pon_marco();
  b1.pon();
  b1.ilumina(3,15);
  _setcursortype(_NOCURSOR);
  a1.escribe(30,4,14,5,"GUIA DE CARRERAS");
  a1.lhor(29,5,47,1,1);
  a1.lver(47,4,4,1,1);
  continua='1';
  raton.inicia();
  xpos=1;
  inicio=ini;
  color=15;  // color del texto
  a1.pon_texto1(8,6,inicio,inicio+13,color);
  _setcursortype(_NOCURSOR);
  raton.inicia();
  continua='1';
  do
  {
    raton.activa();
    if(kbhit())
    {
      car=leetecla();
      if(car=='Q')  // pagina abajo
      {
        if(inicio!=fin-13)
        {
          inicio+=13;
          if(inicio>fin-13)
          {
            inicio=fin-13;
          }
```

```
        raton.desactiva();
        a1.pon_texto1(8,6,inicio,inicio+13,color);
        raton.inicia();
      }
    }
    if(car=='I')  // pagina arriba
    {
      if(inicio+13!=1)
      {
        inicio-=13;
        if(inicio<ini)
        {
          inicio=ini;
        }
        raton.desactiva();
        //a1.borra(6,5,75,20);
        a1.pon_texto1(8,6,inicio,inicio+13,color);
        raton.inicia();
      }
    }
    if(car=='A') // arriba
    {
       if(inicio!=ini)
       {
         --inicio;
         raton.desactiva();
         a1.pon_texto1(8,6,inicio,inicio+13,color);
         raton.inicia();
       }
     }
     if(car=='B')  // abajo
     {
         if(inicio+13!=fin)
         {
           ++inicio;
           raton.desactiva();
           a1.pon_texto1(8,6,inicio,inicio+13,color);
           raton.inicia();
         }
     }
     if(car=='E')  // enter
        {
          if(xpos==1)
          {
            b1.push(3,15);
            delay(100);
            b1.pon();
            raton.desactiva();
            return(1);
          }
        }
  }
  if(raton.boton==1)
    if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // aceptar
       {
         b1.push(3,15);
```

```
            while(raton.boton==1)
              raton.activa();
            b1.pon();
            raton.desactiva();
            return(0);
          }
  } while(continua=='1');
  return(0);
}


int presentadir(char archivo[13], int ini, int fin)
{
  register int xpos, inicio;
  int color;
  char continua, car;
  mouse raton;
  _setcursortype(_NOCURSOR);
  pantalla a1(2,1,77,23,15,5,3,1,archivo);
  botton b1(55,21,70,22,1,3,0,5," ACEPTAR");
  a1.pon_fondo(1,3);
  a1.pon_marco();
  b1.pon();
  b1.ilumina(3,15);
  _setcursortype(_NOCURSOR);
  a1.escribe(25,4,14,5,"DIRECTORIO DE INSTITUCIONES");
  a1.lhor(24,5,52,1,1);
  continua='1';
  raton.inicia();
  xpos=1;
  inicio=ini;
  color=15;  // color del texto
  a1.pon_texto1(8,6,inicio,fin,color);
  _setcursortype(_NOCURSOR);
  raton.inicia();
  continua='1';
  do
  {
   raton.activa();
   if(kbhit())
   {
     car=leetecla();
     if(car=='E')  // enter
         {
           if(xpos==1)
           {
             b1.push(3,15);
             delay(100);
             b1.pon();
             raton.desactiva();
             return(1);
           }
         }
   }
   if(raton.boton==1)
```

```
      if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))  // aceptar
         {
           b1.push(3,15);
           while(raton.boton==1)
             raton.activa();
           b1.pon();
           raton.desactiva();
           return(0);
         }
  } while(continua=='1');
  return(0);
}



int menu_por_carrera(char archi[13], int tam)
{
  typedef struct
  {
    char nombre[66];
    char archivo[13];
    int inicio;
    int fin;
  } carreras;
  carreras carrera[15];
  FILE *fp;
  char car,continua;
  int yini,yfin,actual;
  unsigned char *p;
  register int i,j,xpos,k;
  mouse raton;
  pantalla a1(2,1,77,23,15,5,3,1,"      ");
  botton b1(35,21,50,22,1,3,0,5," MOSTRAR");
  botton b2(55,21,70,22,1,3,0,5,"TERMINAR");
  _setcursortype(_NOCURSOR);
  a1.pon_fondo(1,3);
  a1.pon_marco();
  b1.pon();
  b1.ilumina(3,15);
  b2.pon();
  if(!strcmp(archi,"GCARR18.DAT"))
    a1.escribe(26,3,14,5,"DIRECTORIO DE INSTITUCIONES");
  else
    a1.escribe(31,3,14,5,"GUIA DE CARRERAS");
  a1.lhor(25,4,54,1,1);
  yini=0;
  yfin=14;
  actual=0;
  if((fp=fopen(archi,"r"))==NULL)
  {
    error(1,archi);
    return(0);
  }
  for(i=yini;i<=yfin;i++)
```

```
{
  for(k=0;k<65;k++)
    carrera[i-yini].nombre[k]=' ';
  carrera[i-yini].nombre[65]='\0';
  k=0;
  do
  {
    fscanf(fp,"%c",&car);
    if(car!='\n')
      carrera[i-yini].nombre[k]=car;
    else
      carrera[i-yini].nombre[k]='\0';
    k++;
  } while(car!='\n');
  fscanf(fp,"\n%s\n%d\n%d\n\n",&carrera[i-yini].archivo,
    &carrera[i-yini].inicio,&carrera[i-yini].fin);

  a1.escribe(10,6+i,15,5,carrera[i-yini].nombre);
}
a1.escribe(10,6+actual,11,5,carrera[yini+actual].nombre);
raton.inicia();
actual=0;
continua='1';
xpos=1;
do
{
  raton.activa();
  if(kbhit())
  {
    car=leetecla();
      if(car=='E')  // enter
      {
     if(xpos==1)
     {
          raton.desactiva();
       b2.pon();                                  // CARRERA
          b1.push(3,15);
       delay(100);
       b1.pon();
       b1.ilumina(3,15);
       p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
          salva_pant(0,0,79,24,(char *) p);
       if(!strcmp(archi,"GCARR18.DAT"))
         presentadir(carrera[actual].archivo,carrera[actual].inicio, carrera[actual].fin);
       else
         presenta(carrera[actual].archivo,carrera[actual].inicio, carrera[actual].fin);
          restaura_pant(0,0,79,24,(char *) p);
          free(p);
       raton.inicia();
     }
     else
     {
       raton.desactiva();
       b1.pon();
       b2.push(3,15);
       delay(100);
```

```
      b2.pon();
      continua='0';
    }
   }
 if(car=='Q')  // pagina abajo
 {
   if(yfin!=tam)
   {
     yini+=14;
     yfin+=14;
     if(yfin>tam)
     {
       yfin=tam;
       yini=tam-14;
     }
    // saltar renglones
     fseek(fp,0,0);
     for(i=0;i<yini*5;i++)
     do
       fscanf(fp,"%c",&car);
     while(car!='\n');
     for(i=yini;i<=yfin;i++)
     {
       k=0;
       do
       {
         fscanf(fp,"%c",&car);
         if(car!='\n')
           carrera[i-yini].nombre[k++]=car;
       } while(car!='\n');
       for(j=k;j<65;j++)
         carrera[i-yini].nombre[j]=' ';
       carrera[i-yini].nombre[65]='\0';
       fscanf(fp,"\n%s\n%d\n%d\n\n",&carrera[i-yini].archivo,
         &carrera[i-yini].inicio,&carrera[i-yini].fin);
       a1.escribe(10,6+i-yini,15,5,carrera[i-yini].nombre);
     }
   }
    a1.escribe(10,6+actual,11,5,carrera[actual].nombre);
 }
 if(car=='I')  // pagina arriba
 {
   if(yfin!=0)
   {
     yini-=14;
     yfin-=14;
     if(yini<0)
     {
       yfin=14;
       yini=0;
     }
    // saltar renglones
     fseek(fp,0,0);
     for(i=0;i<yini*5;i++)
     do
       fscanf(fp,"%c",&car);
```

```
    while(car!='\n');
    for(i=yini;i<=yfin;i++)
    {
      k=0;
      do
      {
        fscanf(fp,"%c",&car);
        if(car!='\n')
          carrera[i-yini].nombre[k++]=car;
      } while(car!='\n');
      for(j=k;j<65;j++)
        carrera[i-yini].nombre[j]=' ';
      carrera[i-yini].nombre[65]='\0';
      fscanf(fp,"\n%s\n%d\n%d\n\n",&carrera[i-yini].archivo,
        &carrera[i-yini].inicio,&carrera[i-yini].fin);
      a1.escribe(10,6+i-yini,15,5,carrera[i-yini].nombre);
    }
  }
  a1.escribe(10,6+actual,11,5,carrera[actual].nombre);
}
if(car=='C')  // derecha
  if(xpos==1)
  {
    raton.desactiva();
    b1.pon();
    b2.pon();
    b2.ilumina(3,15);
    xpos=2;
    raton.inicia();
  }
if(car=='D')  // izquierda
  if(xpos==2)
  {
    raton.desactiva();
    b1.pon();
    b2.pon();
    b1.ilumina(3,15);
    raton.inicia();
    xpos=1;
  }
if(car=='A') // arriba
{
  if(actual==0)
  {
    if(yini!=0)
    {
      --yini;
      --yfin;
      // saltar renglones
      fseek(fp,0,0);
      for(i=0;i<yini*5;i++)
      do
        fscanf(fp,"%c",&car);
      while(car!='\n');
      for(i=yini;i<=yfin;i++)
      {
```

```
            k=0;
            do
            {
              fscanf(fp,"%c",&car);
              if(car!='\n')
                carrera[i-yini].nombre[k++]=car;
            } while(car!='\n');
            for(j=k;j<65;j++)
              carrera[i-yini].nombre[j]=' ';
            carrera[i-yini].nombre[65]='\0';
            fscanf(fp,"\n%s\n%d\n%d\n\n",&carrera[i-yini].archivo,
              &carrera[i-yini].inicio,&carrera[i-yini].fin);
            a1.escribe(10,6+i-yini,15,5,carrera[i-yini].nombre);
          }
        }
      }
      else
        --actual;
      a1.escribe(10,6+actual,11,5,carrera[actual].nombre);
      a1.escribe(10,6+actual+1,15,5,carrera[actual+1].nombre);
    }
    if(car=='B') // abajo
    {
      if(actual==14)
      {
        if(yfin!=tam)
        {
          ++yini;
          ++yfin;
          // saltar renglones
          fseek(fp,0,0);
          for(i=0;i<yini*5;i++)
          do
            fscanf(fp,"%c",&car);
          while(car!='\n');
          for(i=yini;i<=yfin;i++)
          {
            k=0;
            do
            {
              fscanf(fp,"%c",&car);
              if(car!='\n')
                carrera[i-yini].nombre[k++]=car;
            } while(car!='\n');
            for(j=k;j<65;j++)
              carrera[i-yini].nombre[j]=' ';
            carrera[i-yini].nombre[65]='\0';
            fscanf(fp,"\n%s\n%d\n%d\n\n",&carrera[i-yini].archivo,
              &carrera[i-yini].inicio,&carrera[i-yini].fin);
            a1.escribe(10,6+i-yini,15,5,carrera[i-yini].nombre);
          }
        }
      }
      else
        if(actual!=tam)
          actual++;
```

```
        a1.escribe(10,6+actual,11,5,carrera[actual].nombre);
        a1.escribe(10,6+actual-1,15,5,carrera[actual-1].nombre);
      }
    }
    if(raton.boton==1)
    {
         if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))   // MOSTRAR
         {
        b2.pon();                                   // CARRERA
          b1.push(3,15);
          while(raton.boton==1)
            raton.activa();
          raton.desactiva();
        b1.pon();
        b1.ilumina(3,15);
        p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
          salva_pant(0,0,79,24,(char *) p);
        if(!strcmp(archi,"GCARR18.DAT"))
          presentadir(carrera[actual].archivo,carrera[actual].inicio, carrera[actual].fin);
         else
          presenta(carrera[actual].archivo,carrera[actual].inicio, carrera[actual].fin);
          restaura_pant(0,0,79,24,(char *) p);
          free(p);
        raton.inicia();
       }
         if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))   // TERMINAR
         {
           b1.pon();
        b2.ilumina(3,15);
          b2.push(3,15);
          while(raton.boton==1)
            raton.activa();
          raton.desactiva();
        b2.pon();
        b2.ilumina(3,15);
        b1.pon();
        continua='0';
         }
    }
  } while(continua=='1');
  fclose(fp);
  return(0);
}



int guia()
{
  typedef struct
  {
    char nombre[61];
    char archivo[13];
    int tam;
  } opciones;
  opciones opcion[9];
```

```
char car,continua;
int yini,yfin,actual;
unsigned char *p;
register int i,j,xpos,k;
mouse raton;
pantalla a1(2,1,77,23,15,5,3,1,"      ");
botton b1(35,21,50,22,1,3,0,5," MOSTRAR");
botton b2(55,21,70,22,1,3,0,5,"TERMINAR");
_setcursortype(_NOCURSOR);
a1.pon_fondo(1,3);
a1.pon_marco();
b1.pon();
b1.ilumina(3,15);
b2.pon();
a1.escribe(31,4,14,5,"GUIA DE CARRERAS");
a1.lhor(25,5,54,1,1);
yini=0;
yfin=14;
actual=0;
strcpy(opcion[0].nombre,"CIENCIAS DE LA SALUD");
strcpy(opcion[1].nombre,"CIENCIAS APLICADAS");
strcpy(opcion[2].nombre,"ADMINISTRACION, FINANZAS Y MERCADOTECNIA");
strcpy(opcion[3].nombre,"CIENCIAS POLITICAS Y SOCIALES");
strcpy(opcion[4].nombre,"CIENCIAS DE LA CONSTRUCCION Y RECURSOS NATURALES");
strcpy(opcion[5].nombre,"CIENCIAS HUMANAS");
strcpy(opcion[6].nombre,"CIENCIAS PURAS");
strcpy(opcion[7].nombre,"CIENCIAS DE LA EDUCACION");
strcpy(opcion[8].nombre,"DIRECTORIO DE INSTITUCIONES");
strcpy(opcion[0].archivo,"GCARR2.DAT"); opcion[0].tam=21;
strcpy(opcion[1].archivo,"GCARR4.DAT"); opcion[1].tam=42;
strcpy(opcion[2].archivo,"GCARR6.DAT"); opcion[2].tam=40;
strcpy(opcion[3].archivo,"GCARR8.DAT"); opcion[3].tam=39;
strcpy(opcion[4].archivo,"GCARR10.DAT"); opcion[4].tam=17;
strcpy(opcion[5].archivo,"GCARR12.DAT"); opcion[5].tam=15;
strcpy(opcion[6].archivo,"GCARR14.DAT"); opcion[6].tam=5;
strcpy(opcion[7].archivo,"GCARR16.DAT"); opcion[7].tam=16;
strcpy(opcion[8].archivo,"GCARR18.DAT"); opcion[8].tam=81;
actual=0;
for(i=0;i<9;i++)
{
  a1.escribe(10,8+i,15,5,opcion[i].nombre);
  a1.escribe(10,8+actual,11,5,opcion[actual].nombre);
}
raton.inicia();
actual=0;
continua='1';
xpos=1;
do
{
  raton.activa();
  if(kbhit())
  {
    car=leetecla();
      if(car=='E')  // enter
      {
    if(xpos==1)
```

```
{
    raton.desactiva();
  b2.pon();                                   // CARRERA
    b1.push(3,15);
  delay(100);
  b1.pon();
  b1.ilumina(3,15);
  p = (unsigned char *) malloc((160) * (25));  /*asignar memoria para el buffer de video*/
    salva_pant(0,0,79,24,(char *) p);
  menu_por_carrera(opcion[actual].archivo,opcion[actual].tam-1);
    restaura_pant(0,0,79,24,(char *) p);
    free(p);
  raton.inicia();
}
else
{
  raton.desactiva();
  b1.pon();
  b2.push(3,15);
  delay(100);
  b2.pon();
  continua='0';
}
 }
if(car=='C')  // derecha
 if(xpos==1)
 {
  raton.desactiva();
  b1.pon();
  b2.pon();
  b2.ilumina(3,15);
  xpos=2;
  raton.inicia();
 }
if(car=='D')  // izquierda
 if(xpos==2)
 {
  raton.desactiva();
  b1.pon();
  b2.pon();
  b1.ilumina(3,15);
  raton.inicia();
  xpos=1;
 }
if(car=='A') // arriba
  if(actual!=0)
  {
   --actual;
   a1.escribe(10,8+actual,11,5,opcion[actual].nombre);
   a1.escribe(10,8+actual+1,15,5,opcion[actual+1].nombre);
  }
 if(car=='B')  // abajo
  if(actual!=8)
  {
   actual++;
   a1.escribe(10,8+actual,11,5,opcion[actual].nombre);
```

```
            a1.escribe(10,8+actual-1,15,5,opcion[actual-1].nombre);
          }
      }
      if(raton.boton==1)
      {
          if((raton.x > 34)&&(raton.x<51)&&(raton.y>21)&&(raton.y<23))   // MOSTRAR
          {
          b2.pon();                                    // CARRERA
             b1.push(3,15);
             while(raton.boton==1)
               raton.activa();
             raton.desactiva();
          b1.pon();
          b1.ilumina(3 15);
          p = (unsigned char *) malloc((160) * (24));  /*asignar memoria para el buffer de video*/
             salva_pant(0,0,79,24,(char *) p);
          menu_por_carrera(opcion[actual].archivo,opcion[actual].tam-1);
             restaura_pant(0,0,79,24,(char *) p);
             free(p);
          raton.inicia();
        }
          if((raton.x > 54)&&(raton.x<71)&&(raton.y>21)&&(raton.y<23))   // TERMINAR
          {
             b1.pon();
          b2.ilumina(3,15);
             b2.push(3,15);
             while(raton.boton==1)
               raton.activa();
             raton.desactiva();
          b2.pon();
          b2.ilumina(3,15);
          b1.pon();
          continua='0';
          }
      }
    } while(continua=='1');
    return(0);
}
```

```
// ARCHIVO MARCO.CPP.
// Es un conjunto de funciones creadas para ayudar a la interface del
// usuario en sus presentaciones y captura de datos.

#include <iostream.h>
#include <dos.h>
#include <conio.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>
#include <bios.h>


void salva_pant(int x1,int y1,int x2,int y2,unsigned char *buffer);
   // guarda una porcion de pantalla en un buffer de memoria
void restaura_pant(int x1, int y1, int x2, int y2, unsigned char *buffer);
   // restaura una porcion de pantalla antes guardada en un buffer
void borra(int x1, int y1, int x2, int y2);
   // borra una porcion de pantalla
char leetecla();  // regresa A-arriba,B-abajo,C-derecha,D-izq.,E-enter,F-esc,G-F1,H-barra
            // Q-pagina abajo, I-pagina arriba
void cambia(int x1, int y1, int x2, int y2, int color, int fondo);
   // cambia una porcion de pantalla
int leenum04(int x, int y, int color, int fondo);
   // lee numeros del 0 al 4
char que_hay(int x, int y);
   // regresa el caracter que hay en x,y en pantalla
int error(int num, char archi[13]);
   // rutina para desplegar mensajes de error


char que_hay(int x, int y)
{
  register int i,j;
  char far *d, far  *mem_vid;
  mem_vid=(char far *) 0xB8000000;
  d=mem_vid + (y-1)*160 + (x-1)*2;  // el primer punto es 0,0
  return(*d);
}


// Rutinas para los botones
class botton {
  int x1,y1,x2,y2;
  int fondo;
  int color;
  int sombra;
  int panta;
  char texto[9];
public:
  botton(int ax1, int ay1, int ax2, int ay2, int afondo, int acolor, int asombra, int apantalla, char
atexto[8]); //constructor
  void push(int color, int tex);
```

```
  void pon();
  void ilumina(int color, int tex);
};

// constructor
botton :: botton(int ax1, int ay1, int ax2, int ay2, int afondo, int acolor, int asombra,int apantalla,
char atexto[9])
{
    x1=ax1;
    y1=ay1;
    x2=ax2;
    y2=ay2;
    fondo=afondo;
    color=acolor;
    sombra=asombra;
    panta=apantalla;
    strcpy(texto,atexto);
    texto[8]='\0';
}


// ilumina el boton
void botton :: ilumina(int color, int tex)
{
  register int i,j;
  int x,y;
  char far *d, far  *mem_vid;
  mem_vid=(char far *) 0xB8000000;
  for(i=y1;i<y2;i++)
    for(j=x1;j<x2;j++)
    {
        d=mem_vid + i*160 + j*2;
        if (*d == 'Ü')
          *++d=(panta*16) | color;
        else
          *++d=(color*16) | tex;
    }
  y=(y2+y1)/2;
  d=mem_vid + y*160 + (x1+1)*2;
  *d++='\x10';
  *d=(color*16) | tex;
  d=mem_vid + y*160 + (x2-2)*2;
  *d++='\x11';
  *d=(color*16) | tex;
}


// pone el boton
void botton :: pon()
{
  register int i,j;
  int x,y;
  char far *d, far  *mem_vid;
  mem_vid=(char far *) 0xB8000000;
  // limpia lo anterior
  for(i=y1-1;i<=y2-1;i++)
```

```
    for(j=x1+2;j<=x2+2;j++)
    {
        d=mem_vid + i*160 + j*2;
        *d++=' ';
        *d=(panta*16) | panta;
    }
  // comienza boton
  for(i=y1;i<y2;i++)
    for(j=x1;j<x2;j++)
    {
        d=mem_vid + i*160 +j*2;
        *d++='Ü';
        *d= (panta*16) | fondo;
    }
    //pone sombra
  for(i=x1+1;i<=x2;i++)
  {
    d=mem_vid + y2*160 + i*2;
    if(i==x2)
        *d++='ß';
    else
        *d++='ß';
    *d=(panta*16) | sombra;
  }
  for(i=y1;i<=y2;i++)
  {
    d=mem_vid + i*160 + x2*2;
    if(i==y1)
        *d++='Ü';
    else
        *d++='Ü';
    *d=(panta*16) | sombra;
  }
  d=mem_vid + y2*160 + x2*2;
  *d++='Ü';
  *d=(sombra*16) | panta;
  // pone texto
  if((y1+1)==y2)
    y=y1;
  else
    y=(y1+y2) / 2;
  x= ((x2-x1-strlen(texto))/2) + x1;
  d=mem_vid + y*160 + x*2;
  for(i=0;i<=strlen(texto);i++)
  {
    *d++=texto[i];
    *d++=(fondo*16) | color;
  }
}


// Pone un boton oprimido
void botton :: push(int color, int tex)
{
  register int i,j;
  unsigned char *p1;
```

```
  char far *d, far *mem_vid;
  ilumina(color,tex);
  mem_vid=(char far *)0xB8000000;                    // abajo agregado
  p1 = (unsigned char *) malloc((x2-x1+3) * (y2-y1+3)); /*asignar memoria para el buffer de
video*/
  salva_pant(x1,y1,x2,y2,p1);
  // borra
  for(i=y1;i<=y2;i++)
    for(j=x1;j<=x2;j++)
    {
        d=mem_vid + i*160 + j*2;
        *d++=' ';
        *d=(panta*16) | panta;
    }
  restaura_pant(x1+2,y1,x2+2,y2,p1);
  free(p1);   // agregado
}


// Rutinas para el manejo del raton
class mouse  {
public:
  int boton;
  int x;
  int y;
  void inicia();
  void activa();
  void desactiva();
};


// Inicializa el raton
void mouse :: inicia()
{
  struct REGPACK reg;
  reg.r_ax =0x001;
  intr(0x33, &reg);
}


// lee las variables del raton
void mouse :: activa()
{
  struct REGPACK reg;
  reg.r_ax=0x003;
  intr(0x33, &reg);
  x=reg.r_cx / 8 +1;
  y=reg.r_dx / 8 +1;
  boton=reg.r_bx;
}


// desactiva el raton
void mouse :: desactiva()
{
```

```
  struct REGPACK reg;
  reg.r_ax=0x002;
  intr(0x33, &reg);
}


// rutinas para el manejo de pantallas
class pantalla  {
  int marco;
  int fondo;
  int sombra,sc;
  char archivo[13];
  int px1;
  int py1, px2,py2;
public:  // constructor
  pantalla(int x1a, int y1a, int x2a, int y2a, int marcoa, int fondoa, int sombraa, int sca, char
archivoa[13]);
  void pon_marco();
  void pon_grueso();
  void pon_delgado();
  void pon_fondo(int color, int ffondo);
  void pon_texto(int x, int y,int inicio, int fin, int color);
  void pon_texto1(int x, int y,int inicio, int fin, int color);
  void escribe(int x, int y, int color, int fondo, char cad[]);
  void pon_paleta(int i);
  void pon_modo(int modo);
  void lver(int x1, int y1, int y2, int color, int grueso);
  void lhor(int x1, int y1, int x2, int color, int grueso);
  void pon_cursor(int cx1, int cy1, int cx2, int cy2, int ccolor, int cfondo);
  void quita_cursor(int cx1, int cy1, int cs2, int cy2);
  void borra(int x1, int y1, int x2, int y2);
};


// pone un cursor
void pantalla :: pon_cursor(int cx1, int cy1, int cx2, int cy2, int ccolor, int cfondo)
{
  register int i,j;
  char far *d, far *mem_vid;
  mem_vid=(char far  *) 0xB8000000;
  for(i=cy1;i<=cy2;i++)
    for(j=cx1;j<=cx2;j++)
    {
        d=mem_vid + i*160 + j*2;
        *++d=(cfondo*16) | ccolor;
    }
}



// quita un cursor
void pantalla :: quita_cursor(int cx1, int cy1, int cx2, int cy2)
{
  register int i,j;
  char far *d, far *mem_vid;
  mem_vid=(char far  *) 0xB8000000;
```

```
  for(i=cy1;i<=cy2;i++)
    for(j=cx1;j<=cx2;j++)
    {
        d=mem_vid + i*160 + j*2;
        *++d=(fondo*16) | 7;
    }
}



// pone el modo de video
void pantalla :: pon_modo(int modo)
{
  union REGS r;
  r.h.al=modo;
  r.h.ah=0;    // funci¢n 0 de la interrupci¢n 16
  int86(0x10, &r, &r);
}

// pone la paleta de colores
void pantalla :: pon_paleta(int i)
{
  union REGS r;
  r.h.bh=1;
  r.h.bl=i;    //n£mero de la paleta
  r.h.ah=0xB;  //funci¢n 11, poner paleta
  int86(0x10, &r, &r);
}


// escribe en pantalla
void pantalla :: escribe(int x, int y, int color, int fondo, char cad[])
{
  gotoxy(x,y);
  textcolor(color);
  textbackground(fondo);
  cprintf(cad);
}

// constructor
pantalla :: pantalla(int x1a, int y1a, int x2a, int y2a, int marcoa, int fondoa, int sombraa, int sca,char
archivoa[])
{
  px1=x1a;
  py1=y1a;
  px2=x2a;
  py2=y2a;
  marco=marcoa;
  fondo=fondoa;
  sombra=sombraa;
  sc=sca;
  strcpy(archivo,archivoa);
  archivo[12]='\0';
}
```

```
// Dibuja un marco
void pantalla :: pon_marco()
{
    register int i,j;
    char far *d, far *c, far *mem_vid;
    mem_vid=(char far*) 0xB8000000;
    for(i=py1;i<=py2;i++)
        for(j=px1;j<=px2;j++)
        {
          d=mem_vid + i*160 + j*2;
          if((i==py1)&&(j==px1))
            *d++='É';
          else
            if((i==py1)&&(j==px2))
              *d++='»';
            else
              if((i==py2)&&(j==px1))
                *d++='È';
              else
                if((i==py2)&&(j==px2))
                  *d++='¼';
                else
                  if((j==px1)||(j==px2))
                    *d++='º';
                  else
                    if((i==py1)||(i==py2))
                      *d++='Í';
                    else
                      *d++=' ';
          *d++= (fondo*16) | marco;
        }
        if(sombra>7)
        {
        // poner sombra
          for(i=py1+1;i<=py2+1;i++)
          {
            d=mem_vid + i*160 + (px2+1)*2;
            *d++='Ü';
            *d++= (fondo*16) | sombra;
            *d++ = 'Ü';
            *d++= (fondo*16) | sombra;
          }
          for(i=px1+1; i<=px2+1; i++)
          {
            d=mem_vid + (py2+1)*160 +i*2;
            *d++ = 'ß';
            *d++ = (fondo*16) | sombra;
          }
        }
        else
        {
          // poner sombra
          for(i=py1+1;i<=py2+1;i++)
          {
            d=mem_vid + i*160 + (px2+1)*2;
```

```
            *++d = (sombra*16) | sc ;
             d++;
            *++d = (sombra*16) | sc;
           }
           for(i=px1+1; i<=px2+1; i++)
           {
             d=mem_vid + (py2+1)*160 +i*2;
             *++d = (sombra*16) | sc;
           }
         }
}



// borra una porcion de pantalla
void pantalla :: borra(int x1, int y1, int x2, int y2)
{
  register int i,j;
  char far *d, far *mem_vid;
  mem_vid=(char far*) 0xB8000000;
  for(i=y1;i<=y2;i++)
    for(j=x1;j<=x2;j++)
    {
      d=mem_vid + i*160 + j*2;
        *d++=' ';
        *d++= (fondo*16) | marco;
    }
}



// Dibuja un marco delgado
void pantalla :: pon_delgado()
{
    register int i,j;
    char far *d, far *c, far *mem_vid;
    mem_vid=(char far*) 0xB8000000;
    for(i=py1;i<=py2;i++)
        for(j=px1;j<=px2;j++)
        {
          d=mem_vid + i*160 + j*2;
          if((i==py1)&&(j==px1))
            *d++='Ú';
          else
            if((i==py1)&&(j==px2))
                *d++='¿';
            else
                if((i==py2)&&(j==px1))
                  *d++='À';
                else
                  if((i==py2)&&(j==px2))
                    *d++='Ù';
                  else
                    if((j==px1)||(j==px2))
```

```
                        *d++='³';
                    else
                        if((i==py1)||(i==py2))
                            *d++='Ä';
                        else
                            *d++=' ';
                *d++= (fondo*16) | marco;
            }
            if(sombra>7)
            {
            // poner sombra
              for(i=py1+1;i<=py2+1;i++)
              {
                d=mem_vid + i*160 + (px2+1)*2;
                *d++='Ü';
                *d++= (fondo*16) | sombra;
                *d++ = 'Ü';
                *d++= (fondo*16) | sombra;
              }
              for(i=px1+1; i<=px2+1; i++)
              {
                d=mem_vid + (py2+1)*160 +i*2;
                *d++ = 'Ü';
                *d++ = (fondo*16) | sombra;
              }
            }
            else
            {
              // poner sombra
              for(i=py1+1;i<=py2+1;i++)
              {
              d=mem_vid + i*160 + (px2+1)*2;
              *++d = (sombra*16) | sc ;
              d++;
              *++d = (sombra*16) | sc;
              }
              for(i=px1+1; i<=px2+1; i++)
              {
                d=mem_vid + (py2+1)*160 +i*2;
                *++d = (sombra*16) | sc;
              }
            }
}


// Dibuja un marco grueo
void pantalla :: pon_grueso()
{
    register int i,j;
    char far *d, far *c, far *mem_vid;
    mem_vid=(char far*) 0xB8000000;
    for(i=py1;i<=py2;i++)
        for(j=px1;j<=px2;j++)
        {
```

```
d=mem_vid + i*160 + j*2;
if((i==py1)&&(j==px1))
  *d++='Ú';
else
  if((i==py1)&&(j==px2))
      *d++='¿';
  else
       if((i==py2)&&(j==px1))
         *d++='À';
       else
         if((i==py2)&&(j==px2))
           *d++='Ù';
         else
           if((j==px1)||(j==px2))
               *d++='³';
           else
               if((i==py1)||(i==py2))
                  *d++='Ä';
               else
                  *d++=' ';
  *d++= (fondo*16) | marco;
}
if(sombra>7)
{
// poner sombra
  for(i=py1+1;i<=py2+1;i++)
  {
    d=mem_vid + i*160 + (px2+1)*2;
    *d++='Ù';
    *d++= (fondo*16) | sombra;
    *d++ = 'Ù';
    *d++= (fondo*16) | sombra;
  }
  for(i=px1+1; i<=px2+1; i++)
  {
    d=mem_vid + (py2+1)*160 +i*2;
    *d++ = 'Ù';
    *d++ = (fondo*16) | sombra;
  }
}
else
{
  // poner sombra
  for(i=py1+1;i<=py2+1;i++)
  {
    d=mem_vid + i*160 + (px2+1)*2;
    *++d = (sombra*16) | sc ;
    d++;
    *++d = (sombra*16) | sc;
  }
  for(i=px1+1; i<=px2+1; i++)
  {
     d=mem_vid + (py2+1)*160 +i*2;
     *++d = (sombra*16) | sc;
  }
}
```

```
}


// Pone un fondo
void pantalla :: pon_fondo(int color, int ffondo)
{
    register int i,j;
    char far *d, far *c, far *mem_vid;
    mem_vid=(char far*) 0xB8000000;
    for(i=0;i<=24;i++)
          for(j=0;j<=79;j++)
          {
            d=mem_vid + i*160 + j*2;
            *d++='';
            *d++= ((ffondo*16) | color;
          }
}


// despliega el contenido de un archivo en pantalla
void pantalla :: pon_texto(int x, int y, int inicio,int fin, int color)
{
  FILE *fp;
  char cad[75], c;
  register int i,j, cuenta;
  textbackground(fondo);
  if((fp=fopen(archivo,"r"))==NULL)  {
    error(1,archivo);
    return;
  }
  // limpiar el arreglo
  for(i=0;i<74;i++)
    cad[i]=' ';
  for(i=1;i<inicio;i++)
    do {
          fscanf(fp,"%c",&c);
    } while(c!='\n');
  cuenta=0;
  for(j=y;j<=20;j++)
  {
    if(!(feof(fp)))
    {
          i=0;
          do {
            fscanf(fp,"%c",&c);
            if(c=='\t')
              c=' ';
            if((c=='\n')||(feof(fp)))
              cad[i]='\0';
            else
              cad[i]=c;
            i++;
          } while((c!='\n')&&(!feof(fp)));
          escribe(x,j,color,fondo,cad);
          cuenta++;
```

```
        if((feof(fp))||(cuenta>=fin))
        j=24;
    }
  }
  fclose(fp);
}

// pone un texto de un archivo
void pantalla :: pon_texto1(int x, int y, int inicio,int fin, int color)
{
  FILE *fp;
  char cad[69], c;
  register int i,j,k, cuenta;
  textbackground(fondo);
  if((fp=fopen(archivo,"r"))==NULL)  {
    error(1,archivo);
    return;
  }
  // limpiar el arreglo
  for(i=1;i<inicio;i++)
    do {
        fscanf(fp,"%c",&c);
    } while(c!='\n');
  cuenta=0;
  for(j=y;j<=20;j++)
  {
    if(!(feof(fp)))
    {
        i=0;
        do {
          fscanf(fp,"%c",&c);
          if(c=='\t')
            c=' ';
          if((c=='\n')||(feof(fp)))
           cad[i]=' ';
          else
            cad[i]=c;
          i++;
        } while((c!='\n')&&(!feof(fp))&&(i<68));
      if(i>68)
        i=68;
      for(k=i;k<68;k++)
        cad[k]=' ';
      cad[68]='\0';
        escribe(x,j,color,fondo,cad);
        cuenta++;
        if((feof(fp))||(cuenta>=fin-inicio+1))
        j=24;
    }
  }
  fclose(fp);
}
```

```
void salva_pant(int x1,int y1,int x2,int y2,unsigned char *buffer)
{
  register int i,j;
  char far *v;
  char far *t;
  v=(char far *) 0xB8000000;
  for(j=y1;j<y2;j++)
    for(i=x1;i<x2;i++)  {
        t=v + (j*160) + i*2;
        *buffer++ = *t++;       /* lee caracter */
        *buffer++=*t;       /* lee atributo */
//      *(t-1)=' ';           /* limpia pantalla */
    }
}



// restaura una porcion de la pantalla
void restaura_pant(int x1, int y1, int x2, int y2, unsigned char *buffer)
{
  register int i,j;
  char far *v, far *t;
  v=(char far *) 0xB8000000;
  t=v;
  for(j=y1;j<y2;j++)
    for(i=x1;i<x2;i++)
    {
        v=t;
        v+= (j*160) + (i*2);    /* calculo de direcci¢n */
        *v++=*buffer++;         /* escribe el caracter */
        *v=*buffer++;
    }
}



void pantalla :: lhor(int x1, int y1, int x2, int color, int grueso)
/* Dibuja una linea horizontal */
{
  register int i;
  textcolor(color);
  textbackground(fondo);
  gotoxy(x1,y1);
  for (i=0;i<=x2-x1;i++)
    if(grueso==0)
        putch('Ä');
    else
        putch('Í');
}


void pantalla :: lver(int x1, int y1, int y2, int color, int grueso)
/* Dibuja una linea vertical */
```

```
{
  register int i;
  char far *t;
  char car;
  textcolor(color);
  textbackground(fondo);
  gotoxy(x1,y1);
  for (i=y1;i<=y2; i++)  {
    gotoxy(x1,i);
    if(grueso==0)
        putch('°');
    else
        putch('Û');
  }
}


void cambia(int x1, int y1, int x2, int y2, int color, int fondo)
{
  register int i,j;
  char far *d, far *mem_vid;
  mem_vid=(char far *) 0xB8000000;
  for(i=y1;i<=y2;i++)
    for(j=x1;j<=x2;j++)
    {
        d=mem_vid + i*160 + j*2;
        if((*d=='[')||(*d==']')||(*d=='(')||(*d==')')||(isdigit(*d)))
          *++d=(fondo*16) | color;
    }
}

void borra(int x1, int y1, int x2, int y2)
{
  register int i,j;
  char far *d, far *mem_vid;
  mem_vid=(char far  *) 0xB8000000;
  for(i=y1;i<=y2;i++)
    for(j=x1;j<=x2;j++)
    {
        d=mem_vid + i*160 + j*2;
        *d++=' ';
    }
}

char leetecla()
{
  char c;
  c=getch();
  if(kbhit())
    c=getch();
  switch(c)
  {
    case 'H':  return('A');   // arriba
    case 'P':  return('B');   // abajo
    case 'M':  return('C');   // derecha
    case 'K':  return('D');   // izquierda
```

```
    case '\r':  return('E');   // enter
    case '\x1B':return('F');   // esc
    case ';' :  return('G');   // F1
    case ' ' :  return('H');   // barra espaciadora
  }
  return(c);
}


int leenum04(int x, int y, int color, int fondo)
{
  char c;
  gotoxy(x,y);
  textcolor(color);
  textbackground(fondo);
  do
  {
    c=getch();
    if((c=='0')||(c=='1')||(c=='2')||(c=='3')||(c=='4'))
        putch(c);
  } while((c!='0')&&(c!='1')&&(c!='2')&&(c!='3')&&(c!='4'));
  return(atoi(&c));
}


// Se define una pantalla emergente para ayuda

class pantemerge {
  int marco;
  int fondo;
  int sombra,sc;
  char archivo[13];
  int px1;
  int py1, px2,py2;
public:  // constructor
  pantemerge(int x1a, int y1a, int x2a, int y2a, int marcoa, int fondoa, int sombraa, int sca, char
archivoa[13]);
  void escribe(int x, int y, int color, int fondo, char cad[]);
  void pon_marco();
  void pon_texto(int inicio, int color);
  int pantemerge :: control(int color);
};

// cosntructor
pantemerge :: pantemerge(int x1a, int y1a, int x2a, int y2a, int marcoa, int fondoa, int sombraa, int
sca,char archivoa[])
{
  px1=x1a;
  py1=y1a;
  px2=x2a;
  py2=y2a;
  marco=marcoa;
  fondo=fondoa;
  sombra=sombraa;
  sc=sca;
  strcpy(archivo,archivoa);
```

```
  archivo[12]='\0';
}


// Dibuja un marco
void pantemerge :: pon_marco()
{
  register int i,j,parx1,parx2,pary1,pary2,k,l;
  char far *d, far *c, far *mem_vid;
  mem_vid=(char far*) 0xB8000000;
  parx1=(px1+px2)/2;
  parx2=(px1+px2)/2;
  pary1=(py1+py2)/2;
  pary2=(py1+py2)/2;
  do
  {
    if(parx1!=px1)
       parx1--;
    if(parx2!=px2)
       parx2++;
    if(parx1!=px1)
       parx1--;
    if(parx2!=px2)
       parx2++;
    if(parx1!=px1)
       parx1--;
    if(parx2!=px2)
       parx2++;
    if(pary1!=py1)
       pary1--;
    if(pary2!=py2)
       pary2++;
    delay(10);
   for(i=pary1;i<=pary2;i++)
    for(j=parx1;j<=parx2;j++)
        {
          d=mem_vid + i*160 + j*2;
          if((i==pary1)&&(j==parx1))
            *d++='É';
          else
            if((i==pary1)&&(j==parx2))
                *d++='»';
             else
                 if((i==pary2)&&(j==parx1))
                   *d++='È';
                 else
                   if((i==pary2)&&(j==parx2))
                     *d++='¼';
                   else
                     if((j==parx1)||(j==parx2))
                         *d++='º';
                     else
                         if((i==pary1)||(i==pary2))
                             *d++='Í';
                         else
                           *d++=' ';
```

```
            *d++= (fondo*16) | marco;
          }
    } while((parx1!=px1)||(parx2!=px2)||(pary1!=py1)||(pary2!=py2));
        if(sombra>7)
        {
        // poner sombra
          for(i=py1+1;i<=py2+1;i++)
          {
           d=mem_vid + i*160 + (px2+1)*2;
           *d++='Ü';
           *d++= (fondo*16) | sombra;
           *d++ = 'Ü';
           *d++= (fondo*16) | sombra;
          }
          for(i=px1+1; i<=px2+1; i++)
          {
            d=mem_vid + (py2+1)*160 +i*2;
            *d++ = 'Ü';
            *d++ = (fondo*16) | sombra;
          }
        }
        else
        {
          // poner sombra
          for(i=py1+1;i<=py2+1;i++)
          {
           d=mem_vid + i*160 + (px2+1)*2;
           *++d = (sombra*16) | sc ;
           d++;
           *++d = (sombra*16) | sc;
          }
          for(i=px1+1; i<=px2+1; i++)
          {
            d=mem_vid + (py2+1)*160 +i*2;
            *++d = (sombra*16) | sc;
          }
        }
      // poner [þ] y -
      textcolor(marco);
      textbackground(fondo);
      gotoxy(px1+3,py1+1);
      cprintf("[ ]");
      textcolor(10);
      gotoxy(px1+4,py1+1);
      cprintf("þ");
      textbackground(3);
      textcolor(1);
      gotoxy(px2+1,py1+2);   cprintf("-");
      gotoxy(px2+1,py2);   cprintf("");
}



// pone el texto en una pantalla emergente
void pantemerge :: pon_texto( int inicio, int color)
```

```
{
  FILE *fp;
  char cad[47], c;
  register int i,j, cuenta,x,y,k;
  textbackground(fondo);
  x=px1+3;
  y=py1+2;
  if((fp=fopen(archivo,"r"))==NULL)  {
    error(1,archivo);
    return;
  }
  // limpiar el arreglo
  for(i=0;i<47;i++)
    cad[i]=' ';
  //saltar renglones
  for(i=1;i<inicio;i++)
    do {
        fscanf(fp,"%c",&c);
    } while((c!='\n')&&!(feof(fp)));
  cuenta=0;
  for(j=inicio;j<=inicio+13;j++)
  {
    if(!(feof(fp)))
    {
        i=0;
        do {
          fscanf(fp,"%c",&c);
          if(c=='\t')
            c=' ';
          if((c=='\n')||(feof(fp)))
           cad[i]=' ';
          else
            cad[i]=c;
          i++;
        } while((c!='\n')&&(!feof(fp)));
      for(k=i;k<46;k++)
        cad[k]=' ';
      cad[46]='\0';
        escribe(x,j+py1+2-inicio,color,fondo,cad);
        cuenta++;
        if((feof(fp))||(cuenta>=inicio+14))
        j=24;
    }
  }
  fclose(fp);
}

// escribe en una pantalla emergente
void pantemerge :: escribe(int x, int y, int color, int fondo, char cad[])
{
  gotoxy(x,y);
  textcolor(color);
  textbackground(fondo);
  cprintf(cad);
}
```

```
// controla una pantalla emergente
int pantemerge :: control(int color)
{
  mouse raton;
  int inicio, fin;
  char continua, car;
  inicio=1;
  if(!strcmp(archivo,"ayuper.txt"))
    fin=245;
  else
    fin=61;  // tama&o del archivo
  pon_texto(inicio,color);
  _setcursortype(_NOCURSOR);
  raton.inicia();
  continua='1';
  do
  {
    raton.activa();
    if(kbhit())
    {
      car=leetecla();
      if(car=='Q')  // pagina abajo
      {
        if(inicio!=fin-13)
        {
          inicio+=13;
          if(inicio>fin-13)
          {
            inicio=fin-13;
          }
          raton.desactiva();
          pon_texto(inicio,color);
          raton.inicia();
        }
      }
      if(car=='I')  // pagina arriba
      {
        if(inicio+13!=1)
        {
          inicio-=13;
          if(inicio<1)
          {
            inicio=1;
          }
          raton.desactiva();
          pon_texto(inicio,color);
          raton.inicia();
        }
      }
      if(car=='A') // arriba
      {
        if(inicio!=1)
        {
          --inicio;
          raton.desactiva();
```

```
            pon_texto(inicio,color);
            raton.inicia();
          }
        }
        if(car=='B')  // abajo
        {
            if(inicio+13!=fin)
            {
              ++inicio;
              raton.desactiva();
              pon_texto(inicio,color);
              raton.inicia();
            }
        }
      }
      if(raton.boton==1)
      {
          if((raton.x==px2+1)&&(raton.y==py2))   // Bajar
          {

            while(raton.boton==1)
          {
            if(inicio+13!=fin)
              {
                ++inicio;
                raton.desactiva();
                pon_texto(inicio,color);
                raton.inicia();
              }
            delay(100);
            raton.activa();
          }
        }
          if((raton.x==px2+1)&&(raton.y==py1+2))   // Subir
          {
          while(raton.boton==1)
          {
            if(inicio!=1)
            {
              --inicio;
              pon_texto(inicio,color);
            }
            delay(100);
            raton.activa();
          }
          }
          if((raton.x== px1+4)&&(raton.y==py1+1))   // Cerrar
          {
          continua='0';
          raton.desactiva();
          return 0;
          }
      }
    } while(continua=='1');
    return(0);
```

```
}


// despliega mensajes de error
int error(int num, char archi[13])
{
  unsigned char *p;
  char continua, car;
  mouse raton;
  botton e1(40,15,55,16,1,3,0,4,"ACEPTAR");
  pantalla a1(13,6,63,17,15,4,0,8,"          ");
  p = (unsigned char *) malloc((160) * (25)); //asignar memoria para el buffer de video
  salva_pant(0,0,79,24,(char *) p);
  _setcursortype(_NOCURSOR);
  a1.pon_marco();
  e1.pon();
  e1.ilumina(3,15);
  switch(num)
  {
    case 1: { a1.escribe(23,10,15,4,"   ERROR AL ABRIR ARCHIVO");
              a1.escribe(32,11,15,4,archi);
            }
            break;
    case 2: a1.escribe(23,10,15,4,"ERROR: NO HAY PAPEL EN IMPRESORA");
            break;
    case 3: a1.escribe(23,10,15,4,"      ERROR EN IMPRESORA");
            break;
    case 4: a1.escribe(23,10,15,4,"ERROR: IMPRESORA FUERA DE LINEA");
            break;
  }
  raton.inicia();
  continua='1';
  do
  {
   raton.activa();
   if(kbhit())
   {
     car=leetecla();
     if(car=='E')
     {
       raton.desactiva();
       e1.push(3,15);
       delay(100);
       e1.pon();
       continua='0';
     }
   }
   if(raton.boton==1)
     if((raton.x > 39)&&(raton.x<56)&&(raton.y>15)&&(raton.y<17))   // aceptar
     {
           e1.push(3,15);
           while(raton.boton==1)
             raton.activa();
           e1.pon();
        e1.ilumina(3,15);
```

```
            raton.desactiva();
        continua='0';
          }
  } while(continua=='1');
  restaura_pant(0,0,79,24,(char *) p);
  free(p);
  return(0);
}
```

Apéndice B

Cuestionarios

CUESTIONARIO DE INTERESES DE HERRERA Y MONTES

Instrucciones: Enseguida encontrarás un total de 60 afirmaciones respecto de diversas actividades. A cada una de ellas le vas a asignar un número que anotarás en el espacio correspondiente de acuerdo a la escala siguiente:

4 significa "me gustaría mucho"
3 significa "me gustaria un poco"
2 significa "me es indiferente"
1 significa "me desagrada un poco"
0 significa "me desagrada mucho"

Al ir respondiendo piensa en la pregunta siguiente:

¿Qué tanto me gustaría...?

[ ] Atender y cuidar enfermos.
[ ] Intervenir activamente en las discusiones en clase.
[ ] Escribir cuentos, crónicas o artículos.
[ ] Dibujar y pintar.
[ ] Cantar en un coro estudiantil.
[ ] Llevar en orden mis libros y cuadernos.
[ ] Conocer y estudiar la estructura de plantas y animales.
[ ] Resolver mecanizaciones numéricas.
[ ] Armar y desarmar objetos mecánicos.
[ ] Salir de excursión.
[ ] Proteger a los muchachos menores del grupo.
[ ] Ser jefe de un grupo.
[ ] Leer obras literarias.
[ ] Moldear el barro, plastilina o cualquier otro material.
[ ] Escuchar música clásica.
[ ] Ordenar y clasificar los libros de una biblioteca.
[ ] Hacer experimentos en un laboratorio.
[ ] Resolver problemas de Aritmética.
[ ] Manejar herramienta, maquinaria.
[ ] Pertenecer a un club de exploradores.
[ ] Ser miembro de un grupo de ayuda y asistencia social.
[ ] Dirigir la campaña política de alguien.
[ ] Hacer versos para una publicación.
[ ] Encargarse del decorado del lugar para un festival.
[ ] Aprender a tocar un instrumento musical.

[ ] Aprender a escribir a máquina y en taquigrafía
[ ] Investigar el origen de las costumbres de los pueblos.
[ ] Llevar las cuentas de una institución
[ ] Construir objetos o muebles
[ ] Trabajar al aire libre fuera de la ciudad
[ ] Enseñar a leer a los analfabetos
[ ] Hacer propaganda para la difusión de una idea.
[ ] Representar un papel en una obra teatral.
[ ] Idear y diseñar el escudo de un club o grupo.
[ ] Ser miembro de un grupo musical.
[ ] Ayudar a calificar pruebas.
[ ] Estudiar y entender las causas de los movimientos sociales.
[ ] Explicar a otros cómo resolver problemas matemáticos.
[ ] Reparar instalaciones eléctricas, de gas o de plomería en casa.
[ ] Sembrar y plantar una granja durante las vacaciones.
[ ] Ayudar a los compañeros en sus dificultades y preocupaciones.
[ ] Leer biografías de políticos eminentes.
[ ] Participar en un concurso de oratoria.
[ ] Diseñar el vestuario para una función teatral.
[ ] Leer biografías de músicos eminentes.
[ ] Encargarse del archivo y los documentos de una asociación.
[ ] Leer libros y revistas científicos.
[ ] Participar en concursos de matemáticas.
[ ] Proyectar y dirigir alguna construcción.
[ ] Atender animales en un rancho durante las vacaciones.

Ahora piensa en esta pregunta:

¿Qué tanto me gustaría trabajar como...?

[ ] Funcionario al servicio de las clases humildes.
[ ] Experto en relaciones p£blicas de una gran empresa.
[ ] Escritor de un peri¢dico o empresa editorial.
[ ] Dibujante profesional en una empresa.
[ ] Concertista en una Sinf¢nica.
[ ] Técnico organizador de oficinas.
[ ] Investigador en un laboratorio.
[ ] Experto calculista en una institución
[ ] Perito mecánico en un gran taller.
[ ] Técnico en el campo, fuera de la ciudad.

CONCENTRADO DE RESULTADOS

Instrucciones.- Se hace un vaciado de los datos en la tabla siguiente, colocando el valor que fue asignado a la pregunta dentro de la casilla correspondiente a su número.

En seguida se suma cada una de las columnas anotando el resultado en el renglón de "Sumas". Por último se obtiene el porcentaje de acuerdo con la equivalencia mostrada.

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60

Sumas									
%									

Suma	%	Suma	%
24	100	10	42
23	96	9	38
22	92	8	33
21	88	7	29
20	83	6	25
19	79	5	21
18	75	4	17
17	71	3	13
16	67	2	6
15	63	1	4
14	58	0	0
13	54		
12	50		
11	46		

INTERPRETACION

Cada una de las columnas de la tabla anterior representa un tipo de interés característico y el porcentaje indica el grado de interés correspondiente. En seguida se muestra la explicación del tipo de interés correspondiente a cada columna:

Columna 1 ASISTENCIAL - Interés por participar en actividades directamente relacionadas con el bienestar de las personas o de la humanidad, tales como, ayudar a otros a resolver sus problemas, informarlos, educarlos,corregirlos, guiarlos o curarlos.

Columna 2. EJECUTIVO-PERSUASIVO.- Interés por dirigir, mandar, planear y organizar las actividades que realizan los otros. Gusto por correr riesgos y aventuras así como tener grandes ambiciones.

Columna 3. VERBAL - Interés por expresarse verbalmente o por escrito. Gusto por la lectura y escritura de cuentos, novelas, diarios , reportajes, poemas, etc.

Columna 4 ARTISTICO-PLASTICO.- Interés por realizar actividades estéticas como el dibujo , la pintura, la escultura, el modelado, el diseño de carteles o letreros, etc, en general la creación de formas y el uso de los colores

Columna 5 MUSICAL - Interés por tocar bien cualquier instrumento; dirigir algún grupo o conjunto, o cantar solo o en coro.

Columna 6 ORGANIZACION.- Interés por ordenar , coleccionar o registrar conjuntos de objetos o números . Gusto por el orden y la sistematización de las cosas.

Columna 7 CIENTIFICO.- Interés por conocer e investigar la razón de ser de los fenómenos, las causas que los provocan y los principios que los explican.

Columna 8 CALCULO.- Interés por resolver problemas numéricos o que involucren ecuaciones matemáticas, ya sea en forma pura o aplicada, por ejemplo, el conteo de dinero (capital e interés), cálculo de superficies y volúmenes, etc.

Columna 9 MECANICO-CONSTRUCTIVO.- Interés por armar, desarmar, construir y ver cómo funcionan aparatos mecánico-eléctricos

Columna 10. ACTIVIDAD AL AIRE LIBRE.- Interés por realizar actividades en lugares abiertos o apartados de los conglomerados urbanos , por ejemplo, cultivar, cuidar ganado, trazar terrenos o ejercicio físico.

CUESTIONARIO DE HABILIDADES DE HERRERA Y MONTES

A continuación te presentamos un listado de 60 afirmaciones, a cada afirmación asígnale un valor en el espacio correspondiente de acuerdo a la escala siguiente:

4 significa "muy hábil"
3 significa "hábil"
2 significa "medianamente hábil"
1 significa "poco hábil"
0 significa "nada hábil"

Al responder piensa en la pregunta siguiente:

¿Qué tan hábil me considero para...? (independientemente de que me guste o no)

[ ] Tratar y hablar con tacto a las personas.
[ ] Ser jefe de un equipo o grupo.
[ ] Expresarme en clase o platicar con mis amigos.
[ ] Dibujar objetos, figuras humanas, etc.
[ ] Cantar en un coro.
[ ] Llevar en forma correcta y ordenada los apuntes de clase.
[ ] Entender principios y experimentos de Biología.
[ ] Ejecutar con rapidez y exactitud operaciones aritméticas.
[ ] Armar y componer objetos mecánicos-eléctricos (planchas, cerraduras, juguetes...)
[ ] Realizar actividades que requieren destreza manual.
[ ] Ser miembro activo y participar en un club social.
[ ] Dirigir y organizar encuentros deportivos, festivales, excursiones o campañas.
[ ] Redactar composiciones o artículos periodísticos.
[ ] Pintar paisajes.
[ ] Aprender a tocar un instrumento musical.
[ ] Ordenar y clasificar debidamente documentos en una oficina.
[ ] Entender principios y experimentos de Física.
[ ] Resolver principios de Aritmética.
[ ] Desarmar, armar y componer objetos complicados.
[ ] Manejar herramienta de carpintería.
[ ] Colaborar con otros para el bien de la comunidad.

[ ] Convencer a otros para que hagan lo que tú crees que se debe hacer.
[ ] Componer versos en serio o en chiste.
[ ] Decorar artísticamente un salón, corredor o patio para un festival.
[ ] Distinguir cuando alguien desentona al cantar o al tocar un instrumento musical.
[ ] Contestar y redactar correctamente cartas y oficios.
[ ] Entender principios y experimentos de Química.
[ ] Resolver rompecabezas de números.
[ ] Armar rompecabezas de alambre o madera.
[ ] Manejar herramientas mecánicas como pinzas, desarmadores o llaves de tuercas.
[ ] Escuchar a otros con paciencia y entender sus puntos de vista.
[ ] Dar órdenes a otros con seguridad y naturalidad.
[ ] Escribir cuentos, narraciones e historietas.
[ ] Modelar con barro, plastilina o grabar madera.
[ ] Aprender a entonar las canciones de moda.
[ ] Manejar y anotar con exactitud y rapidez nombres, números y datos en general.
[ ] Entender principios y hechos económicos y sociales.
[ ] Resolver problemas de Algebra.
[ ] Armar y componer muebles.
[ ] Manejar con habilidad pequeñas piezas herramientas de relojería y joyería.
[ ] Conversar en las reuniones y fiestas con acierto y naturalidad.
[ ] Dirigir un grupo o equipo en situaciones difíciles y peligrosas.
[ ] Saber distinguir y apreciar la buena literatura.
[ ] Saber distinguir y apreciar la buena pintura.
[ ] Saber distinguir y apreciar la buena música.
[ ] Encargarme de recibir, anotar y dar recados sin olvidar detalles importantes.
[ ] Entender las causas que determinan los acontecimientos históricos.
[ ] Resolver problemas de Geometría.
[ ] Aprender el funcionamiento de mecanismos complicados (motores, bombas, etc).
[ ] Hacer trazos geométricos con ayuda de la regla "T" las escuadras y el compás.
[ ] Actuar con desinterés y condolencia.
[ ] Corregir a los demás sin ofenderlos.
[ ] Exponer juicios públicamente sin preocupación de la crítica.
[ ] Colaborar en la elaboración de un libro sobre Arte o Arquitectura.

[ ] Dirigir un conjunto musical.
[ ] Colaborar en el desarrollo de métodos más eficientes de trabajo.
[ ] Realizar investigaciones científicas.
[ ] Enseñar a resolver problemas de Matemáticas.
[ ] Inducir a la gente a obtener resultados prácticos.
[ ] Participar en concursos de modelismo de coches, aviones, barcos, etc.

CONCENTRADO DE RESULTADOS

Instrucciones.- Se hace un vaciado de los datos en la tabla siguiente, colocando el valor que fue asignado a la pregunta dentro de la casilla correspondiente a su número.

En seguida se suma cada una de las columnas anotando el resultado en el renglón de "Sumas". Por último se obtiene el porcentaje de acuerdo con la equivalencia mostrada.

1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60

Sumas									
%									

Suma	%	Suma	%
24	100	10	42
23	96	9	38
22	92	8	33
21	88	7	29
20	83	6	25
19	79	5	21
18	75	4	17
17	71	3	13
16	67	2	6
15	63	1	4
14	58	0	0
13	54		
12	50		
11	46		

INTERPRETACION

Cada una de las columnas de la tabla anterior representa un tipo de habilidad y el porcentaje indica el grado de habilidad correspondiente. En seguida se muestra la explicación del tipo de habilidad correspondiente a cada columna:

Columna 1. SERVICIO SOCIAL.- Habilidad para comprender problemas humanos, para tratar personas, cooperar y preocuparse por los demás. Actitud de ayuda desinteresada hacia sus semejantes.

Columna 2. EJECUTIVO-PERSUASIVO.- Capacidad para organizar, dirigir, supervisar y mandar a otros. Iniciativa, confianza en sí mismo, ambición de progreso. Habilidad para dominar a grupos y personas.

Columna 3. VERBAL.- Habilidad para comprender y expresar con corrección el idioma; para utilizar el lenguaje efectivamente en la comunicación con otros.

Columna 4. ARTISTICO-PLASTICA.- Habilidad para apreciar las formas y los colores de los objetos. Para el dibujo, la escultura, la pintura y el grabado.

Columna 5. MUSICAL.- Habilidad para captar y distinguir sonidos en sus diversas tonalidades, para imaginarlos, reproducirlos, utilizarlos en forma creativa. Sensibilidad en la combinación y armonía de sonidos.

Columna 6. ORGANIZACION.- Habilidad para el orden y la exactitud, rapidez en el manejo de nombres, números, documentos, sistemas y sus detalles en trabajos rutinarios.

Columna 7. CIENTIFICA.- Habilidad para la investigación y capacidad para captar, definir y comprender principios y relaciones causales de los fenómenos, buscando siempre la razón de estos.

Columna 8. CALCULO.- Dominio de las operaciones y mecanizaciones numéricas, así como habilidad para el cálculo matemático.

Columna 9. MECANICA-CONSTRUCTIVA.- Comprensión y habilidad en la manipulación de objetos; facilidad para percibir e imaginar movimientos; así como facilidad para construir o reparar mecanismos.

Columna 10. DESTREZA MANUAL.- Habilidad en el uso de las manos y dedos para el manejo de herramientas finas. Facilidad para realizar trabajos detallados con las manos.

TABLA DE VALORES

Instrucciones.- A continuación encontrarás una lista de valores, lee la descripción de cada uno de ellos y pon una marca en los que son importantes para ti.

( ) AYUDAR A OTROS.- Involucrarse en tareas que beneficien en alguna forma a otros. La sociedad, la comunidad, un grupo de personas o un individuo en particular.

( ) COMPETENCIA.- Participar en actividades en las que pueda mostrar mis habilidades en contra de otros.

( ) CREATIVIDAD ARTISTICA.- Involucrarse en el trabajo creativo en cualquiera de las formas del arte.

( ) TRABAJAR EN GRUPO.-Tener relaciones cercanas de trabajo con otros. Trabajar en equipo.

( ) AVENTURA.- Vivir situaciones que involucran tomar riesgos frecuentemente.

( ) AMISTAD.- Desarrollar relaciones personales cercanas con la gente y hacer amigos.

( ) TRABAJO INDIVIDUAL.- Hacer proyectos individualmente sin importar el contacto con otras personas.

( ) SEGURIDAD. Asegurar un trabajo y una recompensa económica saludable.

( ) ESTETICA.- Involucrarse en el estudio o apreciación de la belleza de las cosas o ideas.

( ) RETOS FISICOS.- Tener un trabajo que requiera demandas de esfuerzo físico mediante los cuales obtenga recompenza.

( ) RIQUEZA.- Llegar a tener mucho dinero y bienes materiales.

( ) ESTABILIDAD.- Tener rutinas de trabajo y obligaciones que sean muy predecibles por períodos largos de tiempo.

( ) CAMBIO Y VARIEDAD.- Tener responsabilidades de trabajo con cambios frecuentes en contenido y ubicación.

( ) TOMAR DECISIONES.- Tener el poder para decidir el curso de acción de las cosas.

( ) CONOCIMIENTO.- Involucrarse en perseguir el conocimiento, la verdad y el entendimiento.

( ) PRECISION EN EL TRABAJO.- Trabajar en situaciones en las cuales hay poca tolerancia al error.

( ) STATUS INTELECTUAL.- Ser reconocido como una persona altamente intelectual o como experta en un campo dado del conocimiento.

( ) TRABAJO PRACTICO.- Les gusta trabajar con cosas concretas, tangibles y de utilidad inmediata.

( ) LIBERTAD DE TIEMPO.- Tener responsabilidades de trabajo que pueda realizar bajo mi propio horario.

( ) AFILIACION.- Ser reconocido como miembro de una organización particular.

( ) CREATIVIDAD CIENTIFICA.- Crear nuevas ideas, estructuras o cualquier otra cosa que siga un formato que no halla sido previamente desarrollado por otros.

TIPOS DE PERSONALIDAD

REALISTA.- Los ejemplos representativos de este tipo, son individuos robustos, prácticos, físicamente fuertes y con frecuencia agresivos en su presencia.

Estas personas tienen usualmente buenas habilidades físicas, mecánico constructivas o de destreza manual, teniendo en cambio problemas para expresarse en palabras o comunicar sus sentimientos a otros.

Sus intereses están en los trabajos al aire libre o en los mecánico-constructivos; les gusta el trabajo con herramientas y maquinaria que implican armar y desarmar cosas; prefieren tener que vérselas con objetos o cosas más bienn que con gente o ideas; son aficionados a los deportes como el fútbol americano y las carreras de automóviles, motos o lanchas.

Sus valores están en relación a las cosas tangibles como la fuerza física y el tener dinero. Les gustan las actividades correctas y prácticas. Generalmente sus ideas políticas y económicas son convencionales y usualmente reacios a las ideas de cambio radical. Les gusta trabajar en lugares abiertos, con mucha frecuencia en el campo o lejos de las zonas urbanas.

Opcionalmente se desenvuelven bien en el área agropecuaria, como ingenieros agrónomos o mecánico electricistas, topógrafos o de construcción pesada y también como técnicos de laboratorio.

INVESTIGATIVO.- Es un tipo de persona que tiende a centrarse alrededor de la ciencia y las actividades científicas y de investigación. Sus habilidades están en el área científica y de cálculo. Tienen facilidad para resolver problemas abstractos; manejar números y ecuaciones matemá ticas; captar la razón de ser de los fenómenos y las causas que los originan. Muestran insuficiencia para las habilidades persuasivas, sociales y rutinarias.

Prefieren pensar en los problemas más que actuar sobre ellos, por lo tanto prefieren trabajar con ideas más que con cosas. Sus áreas de interés son científicas y de cálculo. No les gustan las situaciones con muchas reglas o altamente estructuradas. Trabajan a gusto en cubículos aislados o laboratorios, generalmente en universidades o institutos de investigación.

Frecuentemente tienen valores y actitudes no convencionales, no están particularmente interesados en el dinero, ni en trabajar con otras gentes; valoran mucho la inteligencia, la creatividad científica, el conocimiento y el trabajar individualmente bajo su propio ritmo y horario.

Se desenvuelven bien como físicos, biólogos, químicos, astrónomos, ingenieros en computación, o en otras profesiones que ofrecen posibilidades de investigación como Psicología, Medicina, Economía, etc.

ARTISTICO.- El tipo extremo de estas personas esta orientado artísticamente hacia la autoexpresión de sus sentimientos.

Tienen habilidades en las áreas artístico-plástica, musical o verbal. Aprecian la belleza de las cosas los sonidos o las palabras. Son hábiles en el manejo de las formas y los colores, puntuando muy alto en pruebas de originalidad.

Sus áreas de interés son , artístico-plástico, musical o verbal. Muestran agrado por las manifestaciones estéticas de las cosas, la autoexpresión y el gusto por por la ejecución y el estudio de la música. Prefieren actividades ambiguas, libres y desorganizadas. Tienen muy poco interés en problemas altamante estructurados como el cálculo, o que requieren mucha fortaleza física, prefiriendo trabajar más con ideas o sentimientos que con cosas o datos. Se parecen a las personas clasificadas como investigativas en que prefieren trabajar solos, pero tienen una gran necesidad de expresión individualista , siendo menos asertivos en relación a sus propias capacidades , pero más sensitivos y emocionales.

Se definen a sí mismos omo independientes, originales, no convencionales, expresivos y tensos. Sus valores son, la creatividad artística, la estética, el cambio y la variedad y la independencia. Les gusta trabajar en escenarios donde puedan ser escuchados o admirados, teatros, salas de concierto, galerías de auditorios, medios de información, etc.

Normalmente escogen profesiones como actor, director de teatro, dramaturgo, escritor, reportero, caricaturista, decorador, escultor , músico , escenógrafo , cantante, diseñador o cualquier otra que tenga que ver con el arte.

SOCIAL.- Las personas de este tipo son sociables , responsables y humanistas. Se preocupan por otras personas a quienes les gusta informar, curar, educar o servir de guía.

Muestran habilidades sociales, ubicándose en el área asistencial, son hábiles para comprender problemas humanos, tratar a las personas, cooperar y persuadir.

Prefieren resolver problemas por discusión o mediante arreglos o rearreglos de las relaciones con otros. Por lo tanto les encanta trabajar con personas y no con cosas. Muestran rechazo por las actividades explícitas , ordenadas, sistemáticas y las relacionadas con el uso de instrumentos, materiales o maquinaria. En general su área de interés es la asistencial.

Se describen a sí mismos como animosos, populares, protectores, alegres, sensibles, exitosos y buenos líderes; con perspicacia psicológica y facilidad de palabra. Sus valores son, el ayudar a otros y a la sociedad, el trabajar con otros, la amistad y la afiliación. Buscan ambientes para trabajar en donde puedan relacionarse con otras personas: comunidades, hospitales, escuelas u oficinas públicas.

Se desenvuelven adecuadamente como psicólogos clínicos sociales o educativos, sociólogos, trabajadores sociales, educadores, médicos, licenciados en relaciones públicas, orientadores, y cualquier otra profesión que implique trabajar preferentemente relacionándose con personas.

EMPRENDEDOR.- Las personas emprendedoras prefieren las actividades vinculadas con el manejo de otras personas, para lograr fines organizativos o beneficios económicos.

Poseen una gran facilidad de palabra, la cual usan para vender, dominar o ejercer liderazgo. Tienen capacidad para dirigir, supervisar y mandar a otros adecuadamente. Ambición de progreso, iniciativa y confianza en sí mismos. Sus habilidades en general son del tipo ejecutivo-persuasivo.

Se perciben a sí mismos como enérgicos, entusiastas , aventureros , dominantes, conservadores, impulsivos, estables y deseosos de un alto status. Disfrutan mucho en persuadir a otros sobre sus puntos de vista, se impacientan con el trabajo preciso o con el que involucra largos períodos de esfuerzo intelectual, por lo tanto prefieren trabajar más con personas que con cosas. Su área de interés es la ejecutiva-persuasiva. Les gusta trabajar en ambientes lujosos.

Sus valores están en relación con el poder y la autoridad, la competencia, la toma de decisiones, el dinero y la aventura. Trabajan adecuadamente como vendedores en general, políticos, administradores, gerentes, ejecutivos. En profesiones como Administración de Empresas , Abogado, Lic. en Ciencias Políticas, etc.

CONVENCIONAL.- Las personas convencionales se enfrentan a su ambiente social y físico eligiendo metas, tareas y valores sancionados por la costumbre y la sociedad, enfocando de un modo práctico y correcto los problemas.

Son hábiles para el manejo explícito, ordenado y sistemático de datos, tales como llevar archivos, tomar notas , reproducir materiales , organizar datos escritos y numéricos conforme a un plan prescrito, operar maquinaria de oficina y de procesamiento de datos económicos o de organización. Rechazan las actividades ambiguas , libres, exploratorias o poco sistemáticas. Sus habilidades en general están relacionadas con la organización y el cálculo.

Prefieren las habilidades altamente ordenadas, tanto verbales como numéricas que caracterizan el trabajo de oficina. Se ajustan bien al trabajo en grandes organizaciones y empresas pero no buscan el liderazgo, más bien responden al poder y trabajan a gusto cuando reciben una cadena de instrucciones bien establecidas. Prefieren trabajar con datos concretos más que con ideas abstractas. Sus áreas de interés son Organización y Cálculo.

Se consideran a sí mismos como pulcros, ordenados , inflexibles , conservadores y perseverantes. Sus valores son el dinero y el status, la

estabilidad, la seguridad y la precisión en el trabajo. Trabajan a gusto en oficinas privadas o públicas.

Se desempeñan adecuadamente como contadores públicos, actuarios, operadores de computadora, bibliotecarios, gerentes de banco, analistas de bolsa y finanzas o administradores comerciales.

CATALOGO DE OCUPACIONES

Instrucciones.- A continuación encontrarás un listado de campos ocupacionales agrupados en diez ramas de actividad económica, más una de "Incidencia General". Repasalas y las ramas que más creas que te convienen subráyalas. Ahora busca en detalle los campos ocupacionales que más te interesen y márcalos en el espacio correspondiente.

0.AGRICULTURA, GANADERIA, SILVICULTURA Y PESCA.

( ) 0.1Agricultura.
( ) 0.2Ganadería.
( ) 0.3Silvicultura.
( ) 0.4Pesca.

1.INDUSTRIAS EXTRACTIVAS.

( ) 1.1Extracción y beneficio de carbón mineral, grafito y minerales no metálicos.
( ) 1.2Exploración y explotación de petróleo y gas natural.
( ) 1,3Extracción y beneficio de minerales metálicos.

2.INDUSTRIAS DE TRANSFORMACION. (Alimentos, bebidas, telas, madera, papel, etc.)

( ) 2.0Industria alimentaria.
( ) 2.1Elaboración de bebidas.
( ) 2.2Beneficio y fabricación de productos de tabaco.
( ) 2.3Industria textil.
( ) 2.4Fabricación de prendas de vestir.
( ) 2.5Fabricación de calzado y artículos de cuero y piel.
( ) 2.6Fabricación de productos de aserradero, artículos y accesorios de madera, corcho, palma, vara, carrizo y mimbre.
( ) 2.7Fabricación de muebles y accesorios.
( ) 2.8Industria de papel y cartón.
( ) 2.9Industria editorial, de impresión, encuadernación y actividades conexas.

3.INDUSTRIAS DE LA TRANSFORMACION. (Productos químicos,

metálicos, maquinaria, instrumentos fotográficos, etc.)

( ) 3.0Fabricación de sustancias químicas industriales básicas ; fertilizantes y plaguicidas ; jabones, detergentes , desinfectantes , lustradores y aromatizantes; velas, veladoras y similares;cerillos y fósforos; explosivos y fuegos artificiales.
( ) 3.1Refinación de petróleo y petroquímica básica.
( ) 3.2Fabricación y regeneración de productos de hule y plástico.
( ) 3.3Fabricación de muebles sanitarios, loza, artículos refractarios, de alfarería y cerámica; vidrio y productos de vidrio; productos de arcilla para la construcción; cemento, cal y yeso; abrasivos, productos de mármol y otras piedras; productos y partes preconstruidas de mezclas de cemento con otros materiales;y fabricación de granito artificial y mosaico.
( ) 3.4Industria básica del acero; hierro y metales no ferrosos.
( ) 3.5Fabricación de productos metálicos; utensilios agrícolas, herramientas y artículos de ferrería y cerrajería; muebles metálicos y sus accesorios; estructuras metálicas tanques y similares; envases de hojalata y lámina y otros productos metálicos maquinados.
( ) 3.6Fabricación y ensamble de maquinaria, implementos, equipo y sus partes; máquinas de coser y de oficina.
( ) 3.7Fabricación y ensamble de maquinaria y equipo para generación y transformación de energía eléctrica; equipos y aparatos de radio , televisión y comunicaciones; discos y cintas magnetofónicas; ensamble de aparatos eléctricos y sus partes; acumuladores eléctricos; pilas secas y componentes eléctricos y electrónicos; aparatos y accesorios eléctricos o electrónicos.

( ) 3.8Fabricación y ensamble de automóviles, autobuses, camiones, motocicletas, bicicletas y sus partes; embarcaciones.
( ) 3.9Fabricación y ensamble de aparatos, instrumentos y accesorios de óptica y fotografía; relojes, joyas, artículos de orfebrería y fantasía; instrumentos musicales, paraguas, juguetes, artículos deportivos y otros similares; lápices, gomas, plumas y bolígrafos; talleres de mecánica dental.

4.INDUSTRIA DE LA CONSTRUCCION.

( ) 4.1Construcción de edificaciones y de obras de ingeniería civil.
( ) 4.2Instalaciones sanitarias, eléctricas, de gas y aire acondicionado; ascensores, escaleras móviles y otros equipos para transportación;ventanería, herrería, cancelería, vidrios y cristales; otros servicios vinculados al acabado o remodelación de obras de construcción.

metálicos, maquinaria, instrumentos fotográficos, etc.)

( ) 3.0Fabricación de sustancias químicas industriales básicas ; fertilizantes y plaguicidas ; jabones, detergentes , desinfectantes , lustradores y aromatizantes; velas, veladoras y similares;cerillos y fósforos; explosivos y fuegos artificiales.
( ) 3.1Refinación de petróleo y petroquímica básica.
( ) 3.2Fabricación y regeneración de productos de hule y plástico.
( ) 3.3Fabricación de muebles sanitarios, loza, artículos refractarios, de alfarería y cerámica; vidrio y productos de vidrio; productos de arcilla para la construcción; cemento, cal y yeso; abrasivos, productos de mármol y otras piedras; productos y partes preconstruidas de mezclas de cemento con otros materiales;y fabricación de granito artificial y mosaico.
( ) 3.4Industria básica del acero; hierro y metales no ferrosos.
( ) 3.5Fabricación de productos metálicos; utensilios agrícolas, herramientas y artículos de ferrería y cerrajería; muebles metálicos y sus accesorios; estructuras metálicas tanques y similares; envases de hojalata y lámina y otros productos metálicos maquinados.
( ) 3.6Fabricación y ensamble de maquinaria, implementos, equipo y sus partes; máquinas de coser y de oficina.
( ) 3.7Fabricación y ensamble de maquinaria y equipo para generación y transformación de energía eléctrica; equipos y aparatos de radio , televisión y comunicaciones; discos y cintas magnetofónicas; ensamble de aparatos eléctricos y sus partes; acumuladores eléctricos; pilas secas y componentes eléctricos y electrónicos; aparatos y accesorios eléctricos o electrónicos.

( ) 3.8Fabricación y ensamble de automóviles, autobuses, camiones, motocicletas, bicicletas y sus partes; embarcaciones.
( ) 3.9Fabricación y ensamble de aparatos, instrumentos y accesorios de óptica y fotogrfía; relojes, joyas, artículos de orfebrería y fantasía; instrumentos musicales, paraguas, juguetes, artículos deportivos y otros similares; lápices, gomas, plumas y bolígrafos; talleres de mecánica dental.

4.INDUSTRIA DE LA CONSTRUCCION.

( ) 4.1Construcción de edificaciones y de obras de ingeniería civil.
( ) 4.2Instalaciones sanitarias, eléctricas, de gas y aire acondicionado; ascensores, escaleras móviles y otros equipos para transportación;ventanería, herrería, cancelería, vidrios y cristales; otros servicios vinculados al acabado o remodelación de obras de construcción.

5.INDUSTRIA ELECTRICA.

( ) 5.0Generación, transmisión y distribución de energía eléctrica.

6.COMERCIO.

( ) 6.1Compra-venta de alimentos, bebidas y productos del tabaco.
( ) 6.2Compra-venta de prendas y accesorios de vestir y artículos para su confección; artículos de uso personal; medicamentos, productos farmacéuticos, químico farmacéuticos y de perfumería; papelería, útiles escolares y de oficina, libros, periódicos y revistas.
( ) 6.3Compra-venta de máquinas , muebles, aparatos e instrumentos para el hogar, sus refacciones y accesorios.
( ) 6.4Supermercados , tiendas de autoservicio y de departamentos especializados por línea de mercancía.
( ) 6.5Estaciones de venta de gasolina, diesel y compra-venta de aceites, lubricantes y aditivos.
( ) 6.6Compra-venta de materiales para construcción, madera, acero , productos de ferretería; pieles, cueros curtidos y otros artículos de peletería.
( ) 6.7Compra-venta de maquinaria, equipo y sus refacciones; y accesorios para la producción de bienes y servicios de reparación y mantenimiento.
( ) 6.8Compra-venta de equipo de transporte, refacciones y accesorios.

7.TRANSPORTES Y COMUNICACIONES.

( ) 7.1Autotransportes de pasajeros; transporte ferroviario y eléctrico; metro.
( ) 7.2Transporte marítimo y de navegación interior.
( ) 7.3Transporte aereo.
( ) 7.4Administración de bienes de comunicación; terminales y servicios auxiliares.
( ) 7.5(No registrado en el catálogo).
( ) 7.6Servicios postales, telegráficos y telefónicos.

8.SERVICIOS PARA EMPRESAS, PERSONALES Y EL HOGAR.

( ) 8.1Instituciones de crédito, seguros y fianzas.
( ) 8.2Servicios colaterales a las instituciones financieras y de seguros.
( ) 8.3Servicios relacionados con inmuebles.

( ) 8.4Servicios profesionales y técnicos; de instalación de maquinaria y equipo; de protección y custodia.
( ) 8.5Servicios de alquiler de maquinaria pesada; equipo y mobiliario para empresas y público en general; automóviles, bicicletas, motocicletas y lanchas.
( ) 8.6Servicios de alojamiento temporal y otros servicios auxiliares.
( ) 8.7Preparación y servicio de alimentos y/o bebidas alcohólicas.
( ) 8.8Servicios de esparcimiento.
( ) 8.9Servicios de reparación de artículos de uso doméstico y personal; reparación, lavado y engrasado de vehículos automotores; peluquería y salones de belleza; fumigación, desinfección y control de plagas ; revelado fotográfico ; inhumaciones y servicios conexos; aseo y limpieza.

9.SERVICIOS SOCIALES Y COMUNALES.

( ) 9.1Servicios de enseñanza académica, capacitación, investigación científica y difusión cultural.
( ) 9.2Servicios médicos, paramédicos, auxiliares, de asistencia social y veterinarios.

G.OCUPACIONES DE INCIDENCIA GENERAL.

Este grupo (G) est conformado por ocupaciones de los cuatro niveles (Directivo, Profesional, Técnico y Operativo), que son comunes a dos o más ramas de actividad económica.

( ) G.1Directivo (Ejecutivos Directores, Gerentes, etc.).
( ) G.2Profesional. Investigación (Asesores). Desarrollo (Diseñadores , Ingenieros y Programadores). Producción (Laboratorista de control de calidad). Administración (Auditor, Contador, Programador).
( ) G.3Técnico. Desarrollo (Dibujante). Producción (Inspector, Supervisor). Administración (Almacenista, Auxiliar de Nóminas , Empleado , Operador de Computadora).

AREAS DE ESTUDIO PROFESIONAL

Instrucciones.- A continuación encontrarás un listado de los tipos más representativos de profesionales o técnicos de pais, lo que hacen, su materia de estudio y las materias básicas que se requiere estudiar. Cada uno de ellos representa en realidad un área de estudio especifica, revisa con detalle cada una de ellas y tomando en cuenta tus antecedentes académicos, marca aquellas que consideres más apropiadas para ti.

( ) 1.ABOGADO.- Promueve y defiende juicios en tribunales de justicia. Su materia de estudio es el Derecho. Otras materias: Historia y Sociología.

( ) 2.ACTOR.- Representa papeles dramáticos, cómicos..., en el teatro, cine, televisión, etc. Su materia de estudio es el Arte Dramático. Otras materias: Historia y Literatura.

( ) 3.ADMINISTRADOR.- Planea, organiza, dirige y controla las actividades relacionadas con el personal y las finanzas en empresas públicas y privadas. Su materia de estudio es la Administración. Otras materias: Matemáticas y Derecho.

( ) 4.ANTROPOLOGO.- Estudia e investiga la evolución de las culturas. Su materia de estudio es la Antropología. Otras materias: Historia y Geografía.

( ) 5.ARQUITECTO.- Diseña, proyecta y construye casas, edificios y otras obras. Su materia de estudio es la Arquitectura. Otras materias: Geometría, Dibujo e Historia.

( ) 6.BIBLIOTECOLOGO.- Clasifica, analiza y registra libros, revistas, etc., en una biblioteca o hemeroteca. Su materia de estudio es la Bibliotecología. Otras materias: Historia, Catalogación, Clasificación y Filosofía.

( ) 7.BIOLOGO.- Efectúa estudios integrales sobre los organismos, desde la célula hasta los ecosistemas. Su materia de estudio es la Biología. Otras materias: Química, Matemáticas y Física.

( ) 8.COMUNICOLOGO.- Estudia los procesos de transmisión de mensajes verbales o escritos. Su materia de estudio es la Comunicación. Otras materias: Lenguaje, Psicología e Historia.

( ) 9.CONTADOR.- Formula estados financieros y lleva el control de los ingresos y egresos en establecimientos. Su materia de estudio es la Contabilidad. Otras materias: Matemáticas, Derecho y Economía.

( ) 10.DENTISTA.- Diagnostica, previene y cura afecciones de la boca y los dientes. Su materia de estudio es la Odontología. Otras materias: Anatomía, Química, Biología.

( ) 11.DISEÑADOR.- Idéa y dibuja diversos artículos o material gráfico. Su materia de trabajo es el Diseño. Otras materias: Dibujo, Geometría e Historia del Arte.

( ) 12.ECONOMISTA.- Estudia y analiza los procesos de producción, distribución y consumo de los bienes y servicios del país. Su materia de estudio es la Economía. Otras materias: Historia, Matemáticas y Geografía.

( ) 13.EDUCADOR.- Diseña planes y programas de estudio e imparte mediante diversas técnicas didácticas conocimientos sobre un área específica. Su materia de estudio es la Pedagogía. Otras materias: Filosofía, Psicología y Sociología.

( ) 14.FILOSOFO.- Estudia las corrientes de pensamiento de las diferentes culturas a través de la historia. Su materia de estudio es la Filosofía. Otras materias: Lógica, Historia y Etica.

( ) 15.FISICO.- Estudia los atributos de la materia y la energía. Su materia de estudio es la Física. Otras materias: Matemáticas y Lógica.

( ) 16.GEOGRAFO.- Estudia el uso de los recursos naturales y los problemas poblacionales. Su materia de estudio es la Geografía. Otras materias: Matemáticas, Física y Sociología.

( ) 17.HISTORIADOR.- Analiza y explica los acontecimientos que se han dado a través de la evolución de la humanidad. Su materia de estudio es la Historia. Otras materias: Economía, Sociología y Filosofía.

( ) 18.ING. AGRONOMO.- Estudia las características de las plantas cultivadas, así como los recursos físicos necesarios (suelo, agua, atmósfera), e interviene en los diferentes sistemas de producción agrícola. Su materia de estudio es la Agronomía. Otras materias: Matemáticas, Biología y Química.

( ) 19.ING. CIVIL.- Planea, programa, coordina y supervisa diferentes obras de construcción: puentes, presas, carreteras, vías férreas, estructuras, etc. Su materia de estudio es la Construcción. Otras materias: Matemáticas y Física.

( ) 20.ING. EN COMPUTACION.- Diseña , construye , opera y mantiene sistemas de cómputo. Su materia de estudio es la Programación con Computadora. Otras materias: Matemáticas y Física.

( ) 21.ING. EN ELECTRONICA.- Diseña, construye y mantiene sistemas electrónicos para la industria. Su materia de estudio es la Electrónica. Otras materias: Matemáticas y Física.

( ) 22.ING. INDUSTRIAL .- Planea y organiza sistemas productivos en los que intervienen hombres y máquinas. Su materia de trabajo es la Productividad Industrial. Otras materias: Matemáticas, Física y Economía.

( ) 23.ING. EN INDUSTRIAS EXTRACTIVAS.- (Incluye al Ing. Petrolero, Ing. Geólogo e Ing. Minero). Descubre, evalúa y explota yacimientos minerales, petrolíferos y acuíferos. Su materia de trabajo es el Subsuelo. Otras materias: Química, Física y Matemáticas.

( ) 24.ING. MECANICO ELECTRICISTA .- Diseña y planea instalaciones y máquinas mecánicas y eléctricas; y opera sistemas de generación de energía eléctrica. Sus materias de estudio son la Mecánica y la Electricidad. Otras materias: Física y Matemáticas.

( ) 25.LIC. EN LETRAS.- Estudia la estructura, historia y capacidad de comunicación de diferentes lenguas. Su materia de estudio es la Lengua y la Literatura. Otras materias: Historia, Filosofía y Etimologías.

( ) 26.LIC. EN CIENCIAS POLITICAS.- Analiza el funcionamiento de las instituciones, los partidos políticos, los procesos electorales, etc. Su materia de estudio es la Política. Otras materias: Historia, Economía y Sociología.

( ) 27.LIC. EN RELACIONES PUBLICAS .- (Incluye Lic. en Relaciones Internacionales, Lic. en Relaciones Comerciales y Lic. en Relaciones Industriales). Promueve el diálogo entre grupos de personas, obrero-patrones, entre naciones, entre empresas comerciales o instituciones. Su materia de estudio son las Relaciones Humanas. Otras materias: Derecho , Psicología y Sociología.

( ) 28.LIC. EN TURISMO.- Planea, programa, dirige y supervisa las actividades de un establecimiento dedicado a ofrecer al público servicios de hospedaje,alimentación y entretenimiento. Su materia de estudio es el Turismo. Otras materias: Geografía e Idiomas.

( ) 29.MATEMATICO.- Estudia a profundidad los conceptos y estructuras matemáticas. Puede aplicar sus conocimientos a la solución de problemas en otros campos como la Biología, Química, Física, Astronomía, Economía, etc. Su materia de estudio son las Matemáticas. Otras materias: Idiomas.

( ) 30.MEDICO .- Diagnostica , previene padecimientos y prescribe tratamientos a personas que presentan enfermedades en hospitales, clínicas o consultorios. Su materia de estudio es la Medicina. Otras materias: Anatomía, Biología y Química.

( ) 31.MUSICO.- Estudia la técnica y características melódicas de uno o varios instrumentos e interpreta obras musicales propias o de otros autores. Su materia de estudio es la Música. Otras materias: Solfeo, Armonía y Contrapunto.

( ) 32.NUTRIOLOGO.- Estudia las técnicas de procesamiento y conservación de los alimentos, así como sus propiedades alimenticias. Su materia de estudio es la Nutrición. Otras materias: Biología, Química y Física.

( ) 33.OCEANOLOGO.- Estudia el cuidado y aprovechamiento de los recursos marinos. Su materia de estudio es el Mar. Otras materias: Biología, Química y Física.

( ) 34.PINTOR Y ESCULTOR.- Realiza obras de arte utilizando técnicas de expresión visual para la producción plástica. Su materia de estudio son las Artes Visuales. Otras materias: Dibujo, Historia del Arte y Geometría.

( ) 35.PSICOLOGO.- Estudia los fenómenos del comportamiento humano y los procesos relacionados con éste. Su materia de estudio es la Psicología . Otras materias: Biología , Matemáticas y Lógica.

( ) 36.QUIMICO.- Se encarga de los procesos de transformación de las materias primas en productos y servicios útiles al hombre , en especial a las industrias farmacéutica y de alimentos. Su materia de estudio es la Química. Otras materias: Física, Biología y Matemáticas.

( ) 37.SOCIOLOGO .- Formula estrategias y políticas de desarrollo. Participa en forma interdisciplinaria en la resolución de problemas relacionados con la organización, producción e innovación tecnológica. Su materia de estudio es la Sociedad. Otras materias : Matemáticas , Historia y Filosofía.

( ) 38.TRABAJADOR SOCIAL.- Realiza estudios socioeconómicos en diversos sectores de la población para implementarse en programas de desarrollo comunitario y bienestar social. Su materia de trabajo es el Bienestar Social. Otras materias: Sociología, Psicología y Economía.

( ) 39.TRADUCTOR.- Se dedica a la interpretación y traducción de documentos de diferente índole en uno o varios idiomas. Su materia de estudio son los Idiomas Extranjeros. Otras materias: Lingüística y Gramática.

( ) 40.VETERINARIO.- Aplica técnicas para incrementar la producción de alimentos de origen animal, así como el cuidado de la salud de los animales. Su materia de estudio son los Animales. Otras materias: Biología, Química y Zootecnia.

Apéndice C

Concentrado de Datos sobre Profesiones

PROFESIONES	RAMA Y CAMPO	AREA	PERS. AFIN
ABOGADO	8.4	1	ES
ACTOR	8.8	2	AS
ACTUARIO	8.4	3,20	CI
ADMON. AGROP. Y DES. RURAL	0	3	ER
ADMON. AGROP. Y PESQUERA	0	3	ER
ADMON. BANCARIA	8.1,8.2	3	EC
ADMON. DE CRED. BANCA Y FINAN.	8.1,8.2	3	EC
ADMON. DE EMPRESAS	G	3	ES/EC
ADMON. DE EMPRESAS AGROP.	0	3	ER
ADMON. DE INSTITUCIONES	G	3	EC
ADMON. DE MERCADOTECNIA	6	3	EC
ADMON. DE PERSONAL	G	3	ES
ADMON. DE RECURSOS HUMANOS	G	3	ES
ADMON. DE RECURSOS MARINOS	G	3	ER
ADMON. DE RECURSOS NATURALES	0,1	3	ER
ADMON. DE TIEMPO LIBRE	8.8	27,13	SE
ADMON. EDUCATIVA	9.1	3	SE
ADMON. EMPRESAS MARINAS	G	3	ER
ADMON. EMPRESAS RURALES Y URBANAS	G	3	ES
ADMON. EMPRESAS TURISTICAS	8.6,8.7,8.8	3	ES
ADMON. EMP. TURISTICAS HOTEL. Y REST.	8.6,8.7,8.8	3	ES
ADMON. EMP. TURISTICAS PLANEACION	8.6,8.7,8.8	3	EC
ADMON. FINANCIERA	8.1,8.2	3	EC
ADMON. FISCAL	G	3	CE
ADMON. HOTELERA	8,6	3	ES
ADMON. INDUSTRIAL	2.3	3	EC
ADMON. LABORAL	8.4	3,1	ES
ADMON. LIC EN	G	3	ES/EC
ADMON. PUBLICA	G	3	ES
ADMON. PUBLICA Y CIENC. POLITICAS	G	3	ES
ADMON. TIEMPO LIBRE Y RECREACION	8.8	27,13	SE
ADMON. TURISTICA	8.6,8.7,8.8	3	ES
ADMON. Y FINANZAS	8.1,8.2	3	EC
ADMON. Y ORG. DE EMP. AGROP.	0	3	ER
AGRONOMO TECNICO	0	18	RI
AGROPECUARIO TECNICO	0	18	RI
ANALISIS CLINICOS	G,3	36	SI
ANTROPOLOGIA	9.1	4	IS
ANTROPOLOGIA FISICA	9.1	4	IS
ANTROPOLOGIA SOCIAL	9.1	4	IS
ARCHIVONOMIA	9.1	6	SI
ARQUEOLOGIA	9.1	6	IS
ARQUITECTO	4.1	5	AI
ARQUITECTO NAVAL	G	5	AI
ARQUITECTO URBANISTA	4.1	5	AI
ARQUITECTURA	4.1	5	AI
ARQUITECTURA DEL PAISAJE	4.1	5	AR
ARTES	8.8	2	AS
ARTES MUSICALES	8.8,9.1	31,13	SA
ARTES PLASTICAS	2.9	11	AI
ARTES VISUALES	9.1	34	AS
ARTES VISUALES	9.1	34	AS
ARTES Y DANZA CONTEMPORANEA	8.8	2,3	AS

Apéndice C. Concentrado de Datos sobre Profesiones

PROFESIONES	RAMA Y CAMPO	AREA	PERS. AFIN
AUDICION Y LENGUAJE	9.2	35	SI
BANCA Y FINANZAS	8.1	9	CE
BANCARIO	8.1	9	CE
BIBLIOTECOLOGIA	9.1	6	CS
BIOLOGIA	9.1	7	ES
BIOLOGIA MARINA	0.4	7	IS
BIOLOGIA PESQUERA	0.4	7	IS
BIOLOGO	9.1	7	IS
BIOQUIMICO	9.1	7	IS
CANTO	8.8	31	AS
CIENCIA POLITICA	G.2	26	SI
CIENCIAS ADMINISTRATIVAS	8.1	3	CE
CIENCIAS ATMOSFERICAS	G.2	15	RI
CIENCIAS COMPUTACIONALES	G.2	20	CI
CIENCIAS DE LA COMUNICACION	8.8,2.9	8	AS
CIENCIAS DE LA COMUNIDAD	9	38	RS
CIENCIAS DE LA EDUCACION	9.1	13	SI
CIENCIAS DE LA EDUC. Y CAPAC.	9.1	13	SI
CIENCIAS DE LA INFORMACION	8.8,2.9	8	AS
CIENCIAS DE LA INFORMATICA	G.2	20	CI
CIENCIAS HUMANAS	9.1	25	SI
CIENCIAS JURIDICAS	8.4	1	SE
CIENCIAS POLITICAS	G.2	26	SE
CIENCIAS POLITICAS Y ADMON. PUB.	G.2	26	SE
CIENCIAS QUIMICAS	G.2	36	IR/IS
CIENCIAS SOCIALES	9	38	SI
CIENCIAS Y TEC. DE LA COMUNICACION	8.8,2.9	8	AS
CIENCIAS Y TEC. DE LA INFORMACION	8.8,2.9	8	AS
CIRUJANO DENTISTA	9.2	10	IR
COMERCIALIZACION AGRO-INDUSTRIAL	G.2	9	ER
COMERCIO EXTERIOR	6	27	ES
COMERCIO INTERNACIONAL	6	27	ES
COMPOSICION	8.8	31	AS
COMPUTACION	G.2	20	CI
COMPUTACION ADMINISTRATIVA	G.2	20	CE
COMUNICACION	8.8,2.9	8	AS
COMUNICACION GRAFICA	2.9	11	AS
COMUNICACION HUMANA	8.8,2.9	8	AS
COMUNICACION ORGANIZACIONAL	G.2	27	AS
COMUNICACION SOCIAL	9.1	8	AS
COMUNIC. Y RELACIONES PUBLICAS	G.2	27	SE
CONTADOR PUBLICO	G.2	9	CE
CONTADOR PUBLICO Y AUDITOR	G.2	9	CE
CONTADURIA	G.2	9	CE
CONTADURIA PUBLICA	G.2	9	CE
CONTADURIA PUBLICA Y ADMON.	G.2	9	CE
CONTADURIA Y FINANZAS	G.2	9	CE
CONTROL DE TRANS. AEREO-AERODROMO	7.3,7.4	21	RC
CRIMINALISTICA	8.4	36,30	SI
CRIMINOLOGIA	8.4	1	SI
DEFICIENCIA MENTAL	9.2	13	SI
DEFICIENTES MENTALES	9.2	13	SI
DERECHO	8.4	1	ES
DERECHO Y CIENCIAS SOCIALES	8.4	1,37	ES

PROFESIONES	RAMA Y CAMPO	AREA	PERS. AFIN
DERECHO Y SOCIOLOGIA	8.4	1,37	ES
DESARROLLO DE LA COMUNIDAD	9	37	SR
DESARROLLO ECONOMICO MARITIMO	G.2	12	ER
DIETETICA Y NUTRICION	9.2	32	RS
DISEÑADOR DE LA COMUNICACION GRAFICA	2.9	11	AI
DISEÑADOR DE LOS ASENTAMIENTOS HUMANOS	G.2	5	AI
DISEÑADOR DE MUEBLES	2.7	11	AI
DISEÑADOR DE OBJETOS	G.2	11	AI
DISEÑADOR GRAFICO	2.9	11	AI
DISEÑADOR INDUSTRIAL	G.2	11	AI
DISEÑADOR TEXTIL	2.4	11	AI
DISEÑO	G.2	11	AI
DISEÑO GRAFICO	2.9	11	AI
DISEÑO INDUSTRIAL	G.2	11	AI
DOCENCIA	9.1	13	SI
ECOLOGO MARINO	0.4	7	IR
ECONOMIA	G.2	12	IS
ECONOMIA LABORAL	G.2	12	SI
ECONOMIA POLITICA	G.2	12,26	SE
EDUCACION	9.1	13	SI
EDUCACION BASICA	9.1	13	SI
EDUCACION DE ADULTOS	9.1	13	SI
EDUCACION DE CIEGOS Y DEBILES VISUALES	9.1	13	SI
EDUCACION DE DEFICIENTES MENTALES	9.1	13	SI
EDUCACION DE NIÑOS CON PROBLEMAS APREND.	9.1	13	SI
EDUCACION ESPECIAL	9.1	13	SI
EDUCACION FISICA, DEPORTE Y RECREACION	9.1	13	SR
EDUCACION MEDIA	9.1	13	SI
EDUCACION MEDIA Y SUPERIOR	9.1	13	SI
EDUCACION MUSICAL	9.1	13	SI
EDUCACION PREESCOLAR Y PRIMARIA	9.1	13	SI
EDUCACION TECNOLOGICA	9.1	13	SI
EDUC. DE INADAPTADOS E INFRACTO.	9.1	13	SI
EDUC. DE PERS. CON TRANSTORNOS NEUROTICOS	9.1	13	SI
EDUC. DE PERS. CON TRANSTOR. DE AUDIC. Y LENG.	9.1	13	SI
EDUC. EN DISEÑ. Y SUPERV. DE PROG. INGLES-ESPAÑOL	9.1	13	SI
ELECTRONICA	3.7	21	IR
ENFERMERIA	9.2	30	SC
ENFERMERIA Y OBSTRETICIA	9.2	30	SC
ENSEÑANZA DE IDIOMAS	9.1	13,25	SI
ENSEÑANZA DE INGLES	9.1	13	SI
ENSEÑANZA MEDIA	9.1	13	SI
ENSEÑANZA MUSICAL	9.1	31,13	SA
ESCENOGRAFO	8.8	11,2	AI
ESTADISTICA	G.2	29	IC
ESTADISTICA SOCIAL	G.2	29,37	IC
ESTUDIOS LATINOAMERICANOS	9.1	25	SI
ETNOHISTORIA	9.1	4,17	SI
ETNOLOGIA	9.1	84,17	SI
ETNOMUSICOLOGIA	9.1	31,18	SI
FILOSOFIA	9.1	14	AS
FINANZAS	8.1,8.2	3,4	CE
FISICA	9.1	15	IR
FISICA Y MATEMATICAS	9.1	15,29	IR

Apéndice C. Concentrado de Datos sobre Profesiones

PROFESIONES	RAMA Y CAMPO	AREA	PERS. AFIN
FISICO	9.1	15	IR
FISICO MATEMATICAS	9.1	15	IR
FISICO MATEMATICO	9.1	15	IR
FISIOTERAPIA	9.2	30	SI
GEOGRAFIA	G.2	16	IR
GEOLOGIA MARINA	G.2	23	IR
HISTORIA	9.1	17	SI
HISTORIA DEL ARTE	9.1	17,34	IA
HISTORIA ECONOMICA	9.1	12,17	SI
HOTELERIA Y TURISMO	8.6,8.7,8.8	28	ES
HUMANIDADES	9.1	37	SI
IDIOMA	9.1,8.4	25	SE
INFANTERIA MARINA	7.2	24	RI
INFORMATICA	G.2	20,3	CI
INFORMATICA ADMINISTRATIVA	G.2	20,3	CI
INGENIERIA AGRONOMICA	0.1	18	RI
INGENIERIA BIOMEDICA	9.2	7,30	IS
INGENIERIA BIOQUIMICA	G.2	7,36	IR
INGENIERIA CIVIL	4.1	19	RI
INGENIERIA ELECTRONICA	3.7	21	RI
INGENIERIA EN ELECTRONICA Y DE COMUNICACIONES	3.7,7.6	21	RI
INGENIERIA FISICA	G.2	15	RI
INGENIERIA INDUSTRIAL	3	22	RI
INGENIERIA MECANICA ELECTRICISTA	G.2	24	RI
INGENIERIA MECANICA Y ELECTRICA	G.2	24	RI
INGENIERIA QUIMICA	G.2	36	IR
INGENIERIA TOPOGRAFICA Y GEODESTA	4.1	19	RI
INGENIERO ACUICULTOR	0.4	33	RI
INGENIERO ADMINISTRADOR DE SISTEMAS	G.2	22,3	IE
INGENIERO AGRICOLA	0.1	18	RI
INGENIERO AGRONOMO	0.1	18	RI
INGENIERO AGRONOMO ADMINISTRADOR	G.2	18,3	RE
INGENIERO AGRONOMO AGROINDUSTRIAL	0.3	18	RI
INGENIERO AGRONOMO EN DESARROLLO RURAL	0.1	18	RI
INGENIERO AGRONOMO EN ECONOMIA AGRICOLA	0.1	18	RI
INGENIERO AGRONOMO EN FITOTECNIA	0.1	18	RI
INGENIERO AGRONOMO EN IRRIGACION	0.1	18	RI
INGENIERO AGRONOMO EN PARASITOLOGIA AGRICOLA	0.1	18	RI
INGENIERO AGRONOMO EN PARASITOLOGIA RURAL	0.1	18,37	RI
INGENIERO AGRONOMO EN PRODUCCION	0.1	18	RI
INGENIERO AGRONOMO EN SUELOS	0.1	18	RI
INGENIERO AGRONOMO EN ZONAS ARIDAS	0.1	18	RI
INGENIERO AGRONOMO FITOTECNISTA	0.1	18	RI
INGENIERO AGRONOMO FORESTAL	0.1	18,22	RI
INGENIERO AGRONOMO PARASITOLOGO	0.1	18	RI
INGENIERO AGRONOMO ZOOTECNISTA	0.1,0.2	18,40	RI
INGENIERO AGROQUIMICO	G.2	18,36	RI
INGENIERO AGRO-INDUSTRIAL	G.2	18	RI
INGENIERO ARQUITECTO	4.1	20,5	RI
INGENIERO BIOMEDICO INDUSTRIAL	9.2	7,30	IR
INGENIERO BIOQUIMICO	G.2	7,36	IR
INGENIERO BIOQUIMICO ADMINISTRADOR	G.2	3,7	IE
INGENIERO BIOQUIMICO EN ALIMENTOS	2.0,2.1	7,36	IR
INGENIERO BIOQUIMICO EN PRODUCTOS NATURALES	2.0,2.1,2.2	7,36	IR

PROFESIONES	RAMA Y CAMPO	AREA	PERS. AFIN
INGENIERO CIVIL	4.1	19	RI
INGENIERO CIVIL EN CONSTRUCCION URBANA	4.1	19	RI
INGENIERO CIVIL PORTUARIO	4.1	19	RI
INGENIERO CONSTRUCTOR	4.1	19	RI
INGENIERO DE LOS ALIMENTOS	2.0,2.1	32,36	IR
INGENIERO DE MINAS	1	23	RI
INGENIERO DE MINAS Y METALURGIA	1	23	RI
INGENIERO DE MINAS Y METALURGISTA	1	23	RI
INGENIERO ELECTRICISTA	5	24	RI
INGENIERO ELECTRICISTA ADMINISTRADOR	G.2	24,3	IE
INGENIERO ELECTRICO	5	24	RI
INGENIERO ELECTRICO EN CONTROL	5	24	RI
INGENIERO ELECTRICO EN ELECTRONICA	5	21,24	
INGENIERO ELECTRICO EN POTENCIA	5	24	RI
INGENIERO ELECTROMECANICO	G.2	24	RI
INGENIERO ELECTRONICO EN INSTRUMENTACION	G.2	21	RI
INGENIERO EN AEREONAUTICA	7.3	24	RI
INGENIERO EN AGROINDUSTRIAS	G.2	12	IR
INGENIERO EN AGUAS Y SISTEMAS DE IRRIGACION	4.2	19	RI
INGENIERO EN ALIMENTOS	2.0,2.1	32,22	IR
INGENIERO EN ALIMENTOS MARINOS	2.0	32,22	IR
INGENIERO EN BIOQUIMICA	G.2	7,36	IR
INGENIERO EN CIBERNETICA Y CIENC. DE LA COMP.	G.2	20	IR
INGENIERO EN CIENCIA Y TECNOLOGIA DE ALIMENTOS	2.0,2.1	32	IR
INGENIERO EN CIENCIAS COMPUTACIONALES	G.2	20	IR
INGENIERO EN COMPUTACION	G.2	20	IR
INGENIERO EN COMPUTACION ADMVA. Y DE PRODUCC.	G.2	20	IR
INGENIERO EN COMUNICACIONES Y ELECTRONICA	G.2	21	RI
INGENIERO EN CONSTRUCCION NAVAL	7.2	24	RI
INGENIERO EN CONSTRUCCION URBANA	4.1	19	RI
INGENIERO EN CONTROL E INSTRUMENTACION	G.2	21	RI
INGENIERO EN CONTROL Y COMPUTACION	G.2	21	RI
INGENIERO EN DESARROLLO RURAL	G.2	18,37	RI
INGENIERO EN DISEÑO INDUSTRIAL	G.2	22	RA
INGENIERO EN ELECTROMECANICA	G.2	24	RI
INGENIERO EN ELECTRONICA	G.2	21	RI
INGENIERO EN ELECTRONICA Y COMUNICACIONES	G.2	21	RI
INGENIERO EN ENERGIA	G.2	15,24	RI
INGENIERO EN INDUSTRIAS ALIMENTARIAS	2.0,2.1	32,22	RI
INGENIERO EN IRRIGACION	8.4	19,18	RI
INGENIERO EN MECANICA	G.2	24	RI
INGENIERO EN PESCA INDUSTRIAL	0.4	33,7	RI
INGENIERO EN PLANEACION Y DISEÑO	G.2	22	RI
INGENIERO EN PROCESOS DE ALIMENTOS	2.0,2.1	32,22	RI
INGENIERO EN PROCESOS FARMACEUT. Y BIOTECNOLOG.	3.0	22,36	RI
INGENIERO EN PROCESOS PETROQUIMICOS	3.1	23	RI
INGENIERO EN SISTEMAS COMPUTACIONALES	G.2	20	IR
INGENIERO EN SISTEMAS COMPUTACIONALES EN PROG.	G.2	20	IR
INGENIERO EN SISTEMAS ELECTRICOS Y ELECTRONICOS	G.2	21	RI
INGENIERO EN SISTEMAS ELECTRONICOS	G.2	21	RI
INGENIERO EN SISTEMAS OPERACIONALES	G.2	22	RI
INGENIERO EN TECNOLOGIA DE LA MADERA	2.6	22	RI
INGENIERO EN TRANMISIONES MILITARES	7	21	RI
INGENIERO EN TRANSPORTES	7	24	RI

Apéndice C. Concentrado de Datos sobre Profesiones

PROFESIONES	RAMA Y CAMPO	AREA	PERS. AFIN
INGENIERO EN ZOOTECNIA	0.2	40	RI
INGENIERO FISICO	G.2	15	IR
INGENIERO FISICO INDUSTRIAL	G.2	22	IR
INGENIERO FORESTAL EN SISTEMAS DE PRODUCC.	0.3	18	RI
INGENIERO FRUTICULTOR	0.1	18	RI
INGENIERO GEOFISICO	1	23,15	RI
INGENIERO GEOGRAFO NAVAL	G.2	16	RI
INGENIERO GEOLOGICO	8.4	23	RI
INGENIERO GEOLOGO	8.4	23	RI
INGENIERO GEOLOGO MINERO	1	23	RI
INGENIERO GEOQUIMICO	8.4	23,36	RI
INGENIERO HIDROGRAFO	8.4	23	RI
INGENIERO HIDROLOGO	8.4	18,19	RI
INGENIERO HORTICOLA	0.1	5,18	IR
INGENIERO INDUSTRIAL	G.2	22	RI
INGENIERO INDUSTRIAL ADMINISTRADOR	G.2	22,3	RE
INGENIERO INDUSTRIAL ELECTRICO	G.2	22,24	RI
INGENIERO INDUSTRIAL EN CONTROL CALIDAD	G.2	22	RI
INGENIERO INDUSTRIAL EN ELECTRONICA	G.2	21	RI
INGENIERO INDUSTRIAL EN PLANEACION	G.2	22	RI
INGENIERO INDUSTRIAL EN PRODUCCION	G.2	22	RI
INGENIERO INDUSTRIAL EN QUIMICA	G.2	22,36	RI
INGENIERO INDUSTRIAL EN SIDERURGIA	3.4	23	RI
INGENIERO INDUSTRIAL MECANICO	G.2	22,24	RI
INGENIERO INDUSTRIAL Y DE SISTEMAS	G.2	22	IR
INGENIERO MECANICO	G.2	24	RI
INGENIERO MECANICO ADMINISTRADOR	G.2	24,3	RE
INGENIERO MECANICO AGRICOLA	0.1	18	RI
INGENIERO MECANICO ELECTRICISTA	G.2	24	RI
INGENIERO MECANICO INDUSTRIAL	G.2	24,22	RI
INGENIERO MECANICO METALURGICO	G.2	24,23	RI
INGENIERO MECANICO NAVAL	7.2	24	RI
INGENIERO METALURGICO	3.4	23	RI
INGENIERO METALURGISTA	3.4	23	RI
INGENIERO MINERO	1	23	RI
INGENIERO MINERO Y METALURGISTA	1	23	RI
INGENIERO MUNICIPAL	4.1,4.2	19	RI
INGENIERO NAVAL	7.2	24	RI
INGENIERO PESQUERO	0.4	33,7	RI
INGENIERO PESQUERO EN AGRICULTURA	0.4	33,32	RI
INGENIERO PETROLERO	1.2,3.1	23	RI
INGENIERO PISCICULTOR	0.4	33,7	RI
INGENIERO QUIMICO	G.2	36	RI
INGENIERO QUIMICO ADMINISTRADOR	G.2	36,3	RE
INGENIERO QUIMICO AGROINDUSTRIAL	G.2	36,18	RI
INGENIERO QUIMICO BROMATOLOGO	2.0,2.1	36,18	RI
INGENIERO QUIMICO EN ALIMENTOS	2.0,2.1	36,32	RI
INGENIERO QUIMICO EN PROCESO	G.2	36	RI
INGENIERO QUIMICO INDUSTRIAL	G.2	36,22	RI
INGENIERO QUIMICO METALURGICO	3.4	36,22	RI
INGENIERO QUIMICO PETROLERO	3.1	36,22	RI
INGENIERO QUIMICO Y DE SISTEMAS	G.2	32,22	RI
INGENIERO TEXTIL	2.3	22	RI
INGENIERO TEXTIL EN ACABADOS	2.3	22	RI

PROFESIONES	RAMA Y CAMPO	AREA	PERS. AFIN
INGENIERO TEXTIL EN TEJIDOS DE PUNTOS	2.3	22	RI
INGENIERO TOPOGRAFO	4.1	19	RI
INGENIERO TOPOGRAFO FOTOGRAMETRISTA	8.4	19	RI
INGENIERO TOPOGRAFO GEODESTA	4.1	19	RI
INGENIERO TOPOGRAFO HIDRAULICO	8.4	19	RI
INGENIERO TOPOGRAFO HIDROLOGO	8.4	19	RI
INGENIERO TOPOGRAFO Y FOTOGRAMETRISTA	8.4	19	RI
INGENIERO TOPOGRAFO Y GEODESTA	4.1	19	RI
INGENIERO ZOOTECNISTA	0.2	40	RI
INSTRUMENTACION ELECTRONICA	G.2	21	RI
INSTRUMENTISTA	8.8	31	AS
INTERPRETACION	8.4	39	SC
INVESTIGACION BIOMEDICA BASICA	9.1	7	IR
INVESTIGACION POLITICA	8.4	1	IS
LENGUA INGLESA	9.1	39	AS
LENGUA Y LITERATURAS HISPANICAS	9.1	25	AS
LENGUA Y LITERATURAS MODERNAS	9.1	25	AS
LETRAS	9.1	25	AS
LETRAS CLASICAS	9.1	25	AS
LETRAS ESPAÑOLAS	9.1	25	AS
LETRAS LATINOAMERICANAS	9.1	25	AS
LINGUISTICA	9.1	25,39	SI
LITERATURA	9.1	25	AS
LITERATURA DRAMATICA Y TEATRO	9.1,8.8	2	AS
LITERATURA HISPANICA	9.1	25	AS
LITERATURA IBEROAMERICANA	9.1	25	AS
MANTENIMIENTO DE AEREONAVES Y MOTORES	7.3	24	RI
MANTENIMIENTO DE EQUIPOS ELECTRON. AEREON.	7.3	21	RI
MAQUINISTA NAVAL	7.2	33,24	RI
MATEMATICAS	9.1	29	IC
MATEMATICAS APLICADAS	9.1	29	IC
MATEMATICAS APLICADAS Y COMPUTACION	G.2	29,21	IC
MATEMATICO	9.1	29	IC
MATEMATICO ESTADISTICO	9.1	29	IC
MEDICO CIRUJANO	9.2	30	SI
MEDICO CIRUJANO DENTISTA	9.2	10	SI
MEDICO CIRUJANO Y HOMEOPATA	9.2	30	SI
MEDICO CIRUJANO Y PARTERO	9.2	30	SI
MEDICO ESTOMATOLOGO	9.2	30	SI
MEDICO GENERAL	9.2	30	SI
MEDICO VETERINARIO	0.2,9.2	40	SI
MEDICO VETERINARIO ZOOTECNISTA	0.2,9.2	40	SI
MEDIOS MASIVOS DE LA COMUNICACION	8.8	8	SA
MERCADOTECNIA	6	3	EC
MERCADOTECNIA Y VENTAS	6	3	EC
METEREOLOGIA AEREONAUTICA	7.3	15	RI
NOTARIO	8.4	1	EC
NUTRICION	9.2	32	RI
NUTRICION Y CIENCIA DE LOS ALIMENTOS	9.2	32	RI
OCEANOLOGIA QUIMICA	G.2	33	RI
OCEANOLOGO	8.4	33	IR
ODONTOLOGIA	9.2	10	IR
OFICIAL DE OPERACIONES AERONAUTICAS	7.4	21,15	RI
OPTOMETRIA	3.9	30	IR

Apéndice C. Concentrado de Datos sobre Profesiones

PROFESIONES	RAMA Y CAMPO	AREA	PERS. AFIN
ORGANIZACION DEPORTIVA	9.1	13	SR
PEDAGOGIA	9.1	13	SI
PERIODISMO	2.9	8	SA
PERIODISMO Y COMUNICACION COLECTIVA	2.9	8	SA
PIANO	8.8	31	AS
PILOTO AVIADOR COMERCIAL DE ALA FIJA	7.3	15,24	RI
PILOTO AVIADOR COMERCIAL DE HELICOPTERO	7.3	15,24	RI
PILOTO HELICOPTERISTA NAVAL	7.3	15,24	RI
PILOTO NAVAL	7.2	24,33	RI
PLANIFICACION PARA EL DESARROLLO AGROPEC.	0.1	12,3,4	RI
PROBLEMAS DE APRENDIZAJE	9.1	13,35	SI
PSICOLOGIA	9.2	35	SI
PSICOLOGIA DE LA CONDUCTA SOCIAL	8.4,9.2	35	SI
PSICOLOGIA DEL TRABAJO	8.4	35	SE
PSICOLOGIA EDUCATIVA	8.4,9.2	35	SI
PSICOLOGIA INDUSTRIAL	8.4	35	SE
PSICOLOGIA ORGANIZACIONAL	8.4	35	SE
PSICOLOGIA SOCIAL	8.4,9.2	35	SI
PUBLICIDAD	6	8,27	SE
QUIMICA	G.2	36	IR
QUIMICA FARMACEUTICA BIOLOGA	G.2	36	IR
QUIMICA INDUSTRIAL	G.2	36	IR
QUIMICO	G.2	36	IR
QUIMICO AGRICOLA	0.1	36,18	IR
QUIMICO AGRONOMO	0.1	36,18	IR
QUIMICO BACTERIOLOGO PARASITOLOGO	G.2	36,7	IR
QUIMICO BIOLOGO	G.2	36,7	IR
QUIMICO BIOLOGO AGROPECUARIO	0.1	36,7	IR
QUIMICO BIOLOGO BROMATOLOGO	0.1	36,7	IR
QUIMICO BIOLOGO PARASITOLOGO	0.1	36,7	IR
QUIMICO BROMATOLOGO	0.1	36,7	IR
QUIMICO CLINICO BIOLOGO	G.2	36,7	IR
QUIMICO EN ALIMENTOS	2.0,2.1	36,32	IR
QUIMICO EN METALES	3.4	36,22	IR
QUIMICO FARMACEUTICO BIOLOGO	G.2	36,7	IR
QUIMICO FARMACEUTICO INDUSTRIAL	G.2	36,22	IR
QUIMICO FARMACOBIOLOGO	G.2	36,7	IR
QUIMICO INDUSTRIAL	G.2	36,22	IR
RELACIONES COMERCIALES	6	27	SE
RELACIONES HUMANAS	G.2	27	SE
RELACIONES INDUSTRIALES	G.2	27,22	SE
RELACIONES INTERNACIONALES	8.4	27	SE
RELACIONES PUBLICAS	G.2	27	SE
RELACIONES PUBLICAS Y PUBLICIDAD	G.2	27	SE
RESTAURADOR DE BIENES MUEBLES	8.3	5,17	AI
SALUD PUBLICA	9.2	30	SI
SISTEMAS COMPUTACIONALES	G.2	20	IR
SISTEMAS DE COMPUTACION ADMINISTRATIVA	G.2	20,3	IC
SISTEMAS DIGITALES	G.2	21	IR
SISTEMAS E INFORMATICA	G.2	20,3	IR
SOBRECARGO	7.3	27,28	SE
SOCIOLOGIA	9.1	37	SI
SOCIOLOGIA DE LA EDUCACION	9.1	13,37	SI
TECNOLOGIA Y CIENCIAS QUIMICAS	G.2	36	IR

PROFESIONES	RAMA Y CAMPO	AREA	PERS. AFIN
TEOLOGIA	9.1	14	SA
TERAPIA DE LA AUDICION,VOZ Y EL LENG. ORAL Y ESC.	9.2	30	SI
TERAPIA FISICA	9.2	30	SR
TERAPIA FISICA Y REHABILITACION	9.2	30	SR
TERAPISTA EN PROBLEMAS DE COMUNICACION HUMANA	9.2	35,13	SI
TRABAJO SOCIAL	9.2	38	SI
TRADUCCION	8.4	39	SC
TRAMITACION ADUANAL	8.4	1,9	CR
TRANSTORNOS EN LA AUDICION Y EL LENGUAJE	9.2	13,35	SI
TURISMO	8.6,8.7,8.8	28	ES
TURISMO Y HOTELERIA	8.6,8.7,8.8	28	ES
URBANISMO	4.1	5	AI
VISTA ADUANAL	8.4	1,9	CR
ZOOTECNIA	0.2	40	IR

NOTAS

1. LAS CLAVES DE LA SEGUNDA COLUMNA, "RAMA Y CAMPO", SE DESCRIBEN EN EL APENDICE B EN LA SECCION "CATALOGO DE OCUPACIONES".

2. LA TERCERA COLUMNA, "AREA", SE REFIERE A LAS AREAS DE ESTUDIO Y LAS CLAVES SE DESCRIBEN EN EL APENDICE B EN LA SECCION "AREAS DE ESTUDIO PROFESIONAL".

3. LAS CLAVES DE LA CUARTA COLUMNA, "PERSONALIDAD AFIN", ESTAN EN RELACION CON LOS TIPOS DE PERSONALIDAD DESCRITOS EN EL APENDICE B, LAS CLAVES INDICAN LAS INICIALES DE LOS TIPOS DE PERSONALIDAD, ES DECIR:

R - REALISTA
I - INVESTIGATIVO
A - ARTISTICO
S - SOCIAL
E - EMPRENDEDOR
C - CONVENCIONAL

Bibliografía

Atkinson Lee, Atkinson Mark, *Using Borland C++ 3,* U.S.A., QUE Corporation, 1992, 945 p.

Frenzel, Louis E., Jr., *A Fondo: Sistemas Expertos,* Madrid, Anaya Multimedia, 1987, 213 p.

González Girón, Gilberto, *Un Mundo por Conocer, Manual de Autorientación Vocacional para Alumnos de Bachillerato,* México, Dirección General de Orientación Vocacional, Secretaría de Servicios Académicos, U.N.A.M., 1992. 165 p.

Hayes, John, Hopson, Barrie, *La Orientación Vocacional en la Enseñanza Media,* Londres, Oikos-tau, 1982, 243 p.

Nebendahl, Dieter, *Sistemas Expertos, Introducción a la Técnica y Aplicación,* Marcombo, Siemens, 1990, 225 p.

Rolston, David W., *Principios de Inteligencia Artificial y Sistemas Expertos,* U.S.A., McGraw Hill, 1992, 255 p.

Sánchez y Beltrán, Juan Pablo, *Sistemas Expertos, una Metodología de Programación,* U.S.A., Macrobit, 1990, 261 p.

Schildt Herbert, *C, Guía para Usuarios Expertos,* U.S.A., Osborne / McGraw Hill, 1989, 358 p.

Schildt Herbert, *Turbo C/C++, Manual de Referencia,* U.S.A., McGraw Hill, 1992, 874 p.

Winston, Patrick Henry, *Inteligencia Artificial*, 3a. edición, Massachusetts Institute of Technology, U.S.A., Addison-Wesley Iberoamericana, 1992, 805 p.