



UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE INGENIERIA

ELABORACION DEL SISTEMA "TUTOR" PARA EL EXAMEN DE SELECCION AL CICLO DE BACHILLERATO COMO APLICACION PARA EL AMBIENTE OPERATIVO WINDOWS

T E S I S

QUE PARA OBTENER EL TITULO DE INGENIERO EN COMPUTACION

P R E S E N T A N :

CUAUHTEMOC DOMINGUEZ OLVERA

ARMANDO BARRAGAN MAYA

LUIS ALONSO VASQUEZ GARCIA

ESTELA JIMENEZ CANCHOLA



CIUDAD UNIVERSITARIA NOVIEMBRE DE 1994

TESIS CON FALLA DE ORIGEN



Universidad Nacional
Autónoma de México



UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

DERECHOS RESERVADOS © PROHIBIDA SU REPRODUCCIÓN TOTAL O PARCIAL

Todo el material contenido en esta tesis está protegido por la Ley Federal del Derecho de Autor (LFDA) de los Estados Unidos Mexicanos (México).

El uso de imágenes, fragmentos de videos, y demás material que sea objeto de protección de los derechos de autor, será exclusivamente para fines educativos e informativos y deberá citar la fuente donde la obtuvo mencionando el autor o autores. Cualquier uso distinto como el lucro, reproducción, edición o modificación, será perseguido y sancionado por el respectivo titular de los Derechos de Autor.

Agradecemos a la Universidad Nacional Autónoma de México y en particular a la Facultad de Ingeniería por habernos permitido el paso por sus aulas y haber contribuido a nuestra formación como profesionistas.

Cauhtémoc, Armando, Luis A. , Estela.

Al amor de mi vida
Gabriela Albertos Vivanco

Cuauhtémoc

**A mis Padres,
Ma. de Jesús Cortes de Barragán
L. Jesús Barragán León**

Por su ejemplo, cariño, apoyo y el haberme trazado el camino por el cual, hoy he concluido una etapa de gran importancia en mi vida.

Gracias por Todo.

**A mis Hermanos,
Jesús, Lourdes y Sergio**

Por su apoyo comprensión y cariño mostrado siempre.

A Tí, por haberme dado la vida y estar siempre a mi lado.

A la " Banda " por su apoyo incondicional y ser parte de mi vida.

A la " Banda " por todos los momentos plenos, en los cuales hemos compartido miles de aventuras.

Los quiere

Armando

A mis padres por ser lo que más amo en la vida.

A mis hermanos Luis y Olimpo que siempre me hacen ser feliz y mejor cada día.

A mis amigos de quienes tanto he aprendido y tantas experiencias bellas hemos compartido.

A la UNAM por haberme abierto las puertas a la más valiosa herencia de mi vida.

Luis Alonso

A mi madre, por hacer de mí un ser vivo, un ser humano y la mujer que soy.

A mis hermanos por estar siempre conmigo.

A mis abuelos, por una infancia feliz.

**Gracias a la Universidad, a mis
maestros, a mis compañeros y amigos,
por permitirme incursionar en la
aventura del conocimiento.**

Estela

INDICE

I. INTRODUCCIÓN

- I.1 GENERALIDADES, 1-2**
- 1.2 MARCO TEÓRICO, 2-26**
 - HARDWARE, 3-13**
 - SOFTWARE, 13-20**
 - APLICACIONES, 13**
 - SISTEMAS OPERATIVOS, 13-18**
 - LENGUAJES DE PROGRAMACIÓN, 18-20**
 - RECURSOS HUMANOS, 20-24**
 - BASES DE DATOS, 24-26**
 - DOCUMENTACIÓN, 26**

II. ANÁLISIS

- II.1 DOCUMENTO DE OBJETIVOS DEL PROYECTO, 27-28**
 - IDENTIFICACIÓN Y DEFINICIÓN DE LAS NECESIDADES DEL USUARIO, 27**
 - OBJETIVOS DEL PROYECTO, 27-28**
- II.2 REPORTE DE ANÁLISIS, 28-43**
 - JUSTIFICACIÓN, 28**
 - ALCANCES, 28-33**
 - LIMITACIONES, 33-34**
 - PRUEBA DE FACTIBILIDAD DEL PROYECTO, 34-42**
 - OPERATIVA, 34**
 - TÉCNICA, 34-35**
 - ECONÓMICA, 35-42**
 - MEDIOS, 42**
 - PROPUESTA DEL MÉTODO DE SOLUCIÓN, 42-43**
 - PRIORIDADES, 43**
- II.3 ESPECIFICACIÓN DEL SISTEMA, 44-71**
 - NORMAS DE CALIDAD Y RESULTADOS ESPERADOS, 44-51**
 - DEFINICIÓN DE LA PRECISIÓN DE LOS OBJETIVOS, 51-55**
 - DIAGRAMA DEL SISTEMA, 55-62**
 - SELECCIÓN DEL EQUIPO DE COMPUTO PARA EL DESARROLLO DEL SISTEMA, 62-65**
 - ENTORNO DE PROGRAMACIÓN, 65-70**
 - ENTORNO DEL USUARIO, 70-72**

III. DISEÑO

- III.1 DOCUMENTACIÓN DE LA ARQUITECTURA, 72-79**
 - LOS CONJUNTOS DE INFORMACIÓN, 72-76
 - PROCESOS A DESARROLLAR, 76-78
 - LAS RESPONSABILIDADES, 78-79
- III.2 DISEÑO DEL PROGRAMA DE COMPUTADORA, 80-94**
 - DIAGRAMA DE ESTRUCTURA, 80-88
 - SELECCIÓN DEL LENGUAJE DE PROGRAMACIÓN, 89-94

IV. DESARROLLO

- IV.1 CÓDIGO, 95-99**
 - TÉCNICAS DE PROGRAMACIÓN Y CODIFICACIÓN, 95-99
- IV.2 TÉCNICAS DE PRUEBA DEL SOFTWARE, 99-104**
 - OBJETIVOS DE LA PRUEBA, 99-100
 - DISEÑO DE CASOS DE PRUEBA, 100-104
 - PRUEBA DE LA CAJA BLANCA, 100-101
 - PRUEBA DEL CAMINO BÁSICO, 101
 - PRUEBA DE LA CAJA NEGRA, 101-104
 - PRUEBA DE PARTICIÓN EQUIVALENTE, 102-103
 - ANÁLISIS DE VALORES LÍMITE, 103
 - PRUEBAS DE VALIDACIÓN DE DATOS, 103-104
- IV.3 ESTRATEGIAS DE PRUEBA, 105-110**
 - VERIFICACIÓN Y VALIDACIÓN, 105
 - PRUEBA DE UNIDAD, 105-106
 - PRUEBA DE INTEGRACIÓN, 106-108
 - PRUEBA DE VALIDACIÓN, 108
 - PRUEBAS ALFA Y BETA, 108-109
 - PRUEBA DEL SISTEMA, 109
 - PRUEBA DE DEPURACIÓN, 109-110
- IV.4 PRUEBAS DEL SISTEMA TUTOR, 110-112**
 - PRUEBAS DEL SISTEMA, 110-112

CONCLUSIONES, 114

BIBLIOGRAFÍA

APÉNDICE A CÓDIGO

APÉNDICE B MANUAL TÉCNICO Y CARTA DE NAVEGACIÓN

APÉNDICE C MANUAL DE USUARIO

I. INTRODUCCIÓN

I.1 GENERALIDADES

La Constitución de los Estados Unidos Mexicanos en su artículo 3º, menciona que la educación impartida en territorio nacional es de carácter obligatorio hasta el nivel "Educación Secundaria". Los jóvenes mexicanos que han cumplido con esta obligación y pretenden continuar sus estudios, se ven en la necesidad de prepararse para la realización de un examen de selección de ingreso a escuelas preparatorias públicas o privadas.

La demanda de una oportunidad para ingresar a la Escuela Nacional Preparatoria, CCH (Colegio de ciencias y Humanidades) o Instituciones privadas se ha visto incrementada en los últimos años (de 120,000 solicitudes ingresan sólo 39,000) y por lo tanto el aspirante debe brindar su mejor esfuerzo basándose en los conocimientos adquiridos durante su educación secundaria, haciendo uso de las herramientas que estén a su alcance para prepararse adecuadamente.

Cuando nos mencionan un examen de admisión, en el acto pensamos en lápices del número dos, hojas con largas filas de "ovalitos" para rellenar. Pero los tiempos cambian al igual que los exámenes. En un futuro próximo se estandarizará la realización de exámenes por computadora, siendo IBM de México uno de los precursores. En otros países ya se ha establecido la opción de hacer un gran número de exámenes de admisión por computadora. Por ej. en EE.UU., el ETS (Educational Testing Service) comenzó a ofrecer una versión basada para computadora del GRE (Graduate Record Examination). A finales de 1993, ETS anunció que los exámenes del tipo " papel y lápiz " del GRE serán eliminados próximamente. Al mismo tiempo se anunció que se procedería de igual manera con los exámenes SAT (Schoolastic Assessment Test), LSAT (Law School Admissions Test) y el GMAT (Graduate Managment Admissions Test). Los exámenes mencionados son para nivel Licenciatura y Posgrado. Tomando en cuenta la gran penetración tanto cultural como tecnológica de nuestros vecinos del norte, conviene sensibilizar con este tipo de exámenes a la gente joven de nuestro país. Siendo una gran

oportunidad el desarrollar un sistema basado en computadora con planes de estudios mexicanos y de acuerdo a las exigencias solicitadas por nuestro sistema educacional.

Cuando realizamos un examen de diagnóstico para evaluar nuestro rendimiento en un área específica, estamos propensos a que se cometan errores en la calificación. Pero con el sistema de computadora en segundos se reportarán nuestros aciertos, errores u omisiones. Todo esto para proporcionar un resultado del rendimiento académico.

Es por estas razones que consideramos importante el desarrollar el sistema "Tutor" como una herramienta más para la satisfacción de dicha necesidad.

El sistema pretende cubrir las exigencias académicas de los exámenes tradicionales de admisión. Dentro del tutorial cada examen consiste de un número y tipo de preguntas similar al de un examen real; además de contar con la opción de realizar exámenes por sección que también podrán evaluarse.

Esta es una propuesta formal para realizar un sistema " Tutor " que satisfaga la necesidad de los estudiantes y cumpla con el objetivo de prepararlos para su incorporación al Sistema de Educación Media Superior (Bachillerato).

L2 MARCO TEÓRICO

Descripción de Sistema

- 1. Un conjunto u ordenación de cosas relacionadas de tal manera que forman una unidad o un todo orgánico.**
- 2. Un conjunto de hechos , principios, reglas, etc...clasificadas y ordenadas de tal manera que muestran un plan lógico uniendo las diferentes partes.**
- 3. Un método o plan de clasificación u ordenación.**
- 4. Una forma establecida de hacer algo, un método, un procedimiento...**

[WB94]

A continuación se hará mención de los elementos que definen un sistema basado en computadora (al referirnos a este, lo haremos tan sólo con la palabra sistema).

La elaboración de un sistema requiere:

HARDWARE
SOFTWARE
RECURSOS HUMANOS
BASES DE DATOS
DOCUMENTACIÓN

HARDWARE

Se define como todos los dispositivos electromecánicos y circuitos que constituyen el sistema de computación, cualesquiera partes físicas del sistema incluyendo circuitos integrados, terminales de video, impresoras, dispositivos auxiliares de memoria y todo tipo de dispositivos periféricos.

Del ábaco a la computadora personal

El ábaco y la computadora personal son dos pequeños dispositivos para proceso de datos separados por miles de años de historia. Los siguientes son algunos de los avances más importantes en este largo camino.

Los primeros archivistas y sus instrumentos

Durante miles de años los seres humanos vivieron sobre la Tierra sin registrar la información en forma permanente. Sin embargo, al crecer las tribus y formar naciones, se desarrolló el comercio. Alrededor de 3500 a. C., los mercaderes babilónicos mantenían archivos en tableros de barro. El ábaco fue uno de los primeros instrumentos manuales para calcular, y aunque tiene una antigüedad de más de 3000 años, se sigue utilizando en algunas partes del mundo.

Las técnicas de archivado se siguieron desarrollando a través de los siglos, presentándose innovaciones como las auditorías (los griegos) y los presupuestos (los romanos). En 1642 Blas Pascal, un brillante joven francés, desarrolló la primera máquina calculadora mecánica. Unos treinta años más tarde, el matemático alemán Gottfried Von Leibniz mejoró el invento de Pascal produciendo una máquina que podía sumar, restar, multiplicar, dividir y extraer raíces. Sin embargo, no existía la tecnología para fabricar este tipo de instrumentos de precisión en forma masiva.

Todavía en la década de 1880, el procesamiento de datos en Estados Unidos se llevaba a cabo con lápices, plumas y reglas. El volumen de los archivos durante este periodo estaba creciendo rápidamente y, como era de esperarse, los métodos manuales producían información relevante inexacta y muchas veces tardía. Para desesperación de la oficina de censos de Estados Unidos, por ejemplo, el censo de 1880 no se terminó hasta cuando comenzaba el censo de 1890. Por fortuna para esta oficina y para otros que requerían mejores métodos de procesamiento de datos, por esas fechas se inventó el equipo electromecánico para trabajar con tarjetas perforadas.

El tejedor, el estadístico y sus tarjetas

La historia de las tarjetas perforadas data de 1801, cuando un tejedor francés llamado Joseph Marie Jacquard las inventó para controlar sus telares mecánicos. No obstante, lo que hizo que se utilizaran como medio para el procesamiento de datos fue el problema de completar el censo de 1880. El inventor de las técnicas de tarjetas perforadas fue el doctor Herman Hollerith, un estadístico. La oficina de censos lo contrató para encontrar una solución al problema del censo. En 1887, Hollerith desarrolló su concepto de tarjeta de lectura mecánica y diseñó un dispositivo conocido como la "máquina del censo" . La tabulación con los métodos de Hollerith requería una octava parte del tiempo que se necesitaba antes, por lo que se adoptaron sus técnicas para el censo de 1890. Aunque la población había aumentado de 50 a 63 millones en la década posterior a 1880, el censo se completó en menos de tres años (naturalmente, esto representa una lentitud intolerable para la época actual. El censo de 1950, utilizando equipo de tarjetas perforadas,

requirió casi dos años; el censo computarizado de 1980 produjo cifras totales en unos cuatro meses).

Después del censo de 1890, Hollerith adaptó su equipo para utilizarlo comercialmente y creó sistemas de estadística de carga para dos ferrocarriles. En 1896 fundó la Tabulating Machine Company para fabricar y vender su invento. Más tarde, esta firma se fusionó con otras para formar la International Business Machines (IBM) Corporation.

El procesamiento de tarjetas perforadas se basó en una idea sencilla: los datos de entrada se registran primero en forma codificada haciendo perforaciones en tarjetas; más tarde las tarjetas se introducen en una serie de máquinas electromecánicas que realizan los pasos de procesamiento el cual era mucho más rápido y exacto que los métodos manuales. No obstante, requería todavía personas que llevaran las cajas con tarjetas de un paso al siguiente. Era necesario alimentar cada una de las máquinas, echarla a andar y detenerla. La necesidad de intervención humana era una desventaja importante. Con la computadora, naturalmente, esa desventaja desaparece. No se requiere intervención humana entre los distintos pasos.

Avances iniciales en el desarrollo de las computadoras

“ La locura de Babbage” Unos cincuenta años antes de los trabajos de Hollerith, Charles Babbage, profesor de matemáticas en la Cambridge University, de Inglaterra, propuso una máquina a la que le dio el nombre de “máquina analítica”. Babbage era un hombre excéntrico y de mal carácter que pasó gran parte de su vida trabajando en vano para completar su increíblemente compleja máquina. El sueño de Babbage hubiera incluido una entrada por tarjetas perforadas, una unidad de memoria o almacén, una unidad aritmética o molino, la impresión automática de salida, el control secuencial por programa y una exactitud de veinte cifras. En pocas palabras, Babbage había diseñado un prototipo de computadora que estaba adelantada en cien años a su época. Lady Augusta Ada Lovelace, hija de Lord Byron, el poeta, ayudó a Babbage. Ella era una brillante matemática y corrigió algunos errores en el trabajo de Babbage e inventó formas novedosas de enfocar el diseño de programas empleando tarjetas perforadas. Al morir Babbage, en

1871, el desarrollo de las computadoras se detuvo hasta 1937; las tarjetas perforadas dominaron el mundo del proceso de datos.

ABC, ENIAC y los primeros pioneros

En 1937, Howard Aiken, profesor de Harvard, se fijó la meta de construir una máquina calculadora automática que combinara la tecnología eléctrica y mecánica con las técnicas de tarjetas perforadas de Hollerith. Con la ayuda de estudiantes de posgrado e ingenieros de IBM, el proyecto se completó en 1944. El aparato terminado se denominó la computadora digital Mark I. Las operaciones internas se controlaban automáticamente con relevadores electromagnéticos, y los contadores aritméticos eran mecánicos; así, la Mark I era una computadora electromecánica. En muchos aspectos era el sueño de Babbage hecho realidad. Esta máquina "medieval" se exhibe actualmente en la Universidad de Harvard.

El primer prototipo de computadora electrónica se concibió en el invierno de 1937-1938 por el doctor John Vincent Atanasoff, profesor de física y matemáticas en Iowa State College. Atanasoff formó un equipo con Clifford Berry, su asistente de posgrado, y comenzó a construir la primera computadora electrónica. La llamaron "Computadora Atanasoff - Berry", o ABC. La ABC empleaba bulbos al vacío para almacenar datos y efectuar operaciones aritméticas y lógicas.

Durante 1940 y 1941 Atanasoff y Berry se reunieron con John W. Mauchly y le mostraron su trabajo. Mauchly, que trabajaba en la School of Electrical Engineering de la University of Pennsylvania, comenzó a pensar en la forma de construir una computadora de aplicación general. (La ABC se diseñó con el objetivo específico de resolver sistemas de ecuaciones simultáneas). Mauchly formó un equipo con J. Presper Eckert, estudiante de posgrado de Ingeniería en la Moore School, para organizar la construcción de ENIAC a principios de la década de 1940.

ENIAC fue la primera computadora electrónica de aplicación general que entró en funcionamiento. Financiado por el ejército de los Estados Unidos, se construyó en la Moore School como proyecto secreto durante la guerra (al

ejército le interesaba la preparación rápida de tablas de trayectorias de proyectiles). También se utilizaron bulbos al vacío (18,000 de ellos) en ENIAC. Aunque pesaba 30 toneladas y ocupaba el espacio de una casa de tres recámaras, ENIAC podría hacer 300 multiplicaciones por segundo, lo que la hacía 300 veces más rápida que cualquier otro dispositivo de la época. Las instrucciones de operación de ENIAC no se almacenaban internamente, más bien se introducían por medio de tableros de clavijas e interruptores localizados en el exterior. El ejército utilizó la ENIAC hasta 1955 y después se colocó en el Smithsonian Institution.

A mediados de la década de 1940 colaborando con H.L.Goldstine y A.W.Burks, el genio matemático John Von Neumann escribió un trabajo sugiriendo:

- 1) Utilizar sistemas de numeración binarios para construir computadoras.
- 2) Que las instrucciones para la computadora, así como los datos que se manipulan, se almacenaran internamente en la máquina.

La primera de esas ideas se había utilizado ya en la ABC de Atanasoff; el sistema de numeración binario utiliza únicamente dos dígitos (0 y 1) en vez de los diez dígitos (0 a 9) del sistema decimal con el que todo mundo estamos familiarizados. Dado que los componentes electrónicos están normalmente en uno de los dos estados ("encendido" o "apagado"), el concepto binario simplificó el diseño del equipo.

La segunda idea fue un avance brillante en esa época porque significaba que los programas se podían almacenar de la misma forma cómo se almacenaban los datos. Ya no había necesidad de cambiar un sinnúmero de interruptores y cables antes de ejecutar una aplicación nueva.

Aunque estas ideas no se incorporaron en ENIAC, Maunchly, Eckert y otros de Moore School se dedicaron a la tarea de construir una máquina que pudiera almacenar programas. Esta máquina, la EDVAC, no se terminó sino hasta varios años más tarde. La distinción de ser la primera computadora electrónica de programa almacenado corresponde a la EDSAC, terminada en 1949 en Cambridge University.

La carrera para entrar al mercado

Una de las razones del retraso sufrido por EDVAC fue que Eckert y Mauchly fundaron su propia compañía en 1946 y comenzaron a trabajar en la Universal Automatic Computer, o UNIVAC. A principios de 1951 la primera UNIVAC-1 comenzó a funcionar en la oficina del censo. La IBM 650 entró en servicio por primera vez en Boston a fines de 1954, siendo una máquina relativamente barata para aquella época, tuvo gran aceptación, y dio a la IBM liderazgo en la producción de computadoras en 1955.

En el periodo de 1954 a 1959, muchos negocios adquirieron computadoras para procesar datos, aún cuando estas máquinas de la primera generación habían sido diseñadas para aplicaciones científicas. Los usuarios no científicos solían considerar la computadora como un instrumento de contabilidad y las primeras aplicaciones de negocios se diseñaron para procesar tareas rutinarias como son las nóminas. Se subestimó el potencial real de las computadoras y muchas fueron adquiridas por el prestigio que conferían a la organización.

La segunda y tercera generación

La segunda generación comenzó en 1959, eran más pequeñas y rápidas y tenían una capacidad de cómputo mayor. La escritura de programas de aplicación en lenguaje de máquina fue desplazada por el uso de lenguaje de programación de alto nivel y el bulbo al vacío, de vida relativamente corta, fue desplazado por componentes compactos de estado sólido, como los transistores que habían sido desarrollados en los Bell Laboratories en 1947 por John Bardeen, William Shockley y Walter Brattain.

Al mismo tiempo que se desarrollaban los sistemas de la segunda generación se estaba creando una industria nueva, basada en la idea de integrar transistores y otros componentes para formar circuitos que pudieran colocarse en pequeños trozos de silicio. Una de las primeras compañías de esta industria fue Shockley Semiconductor, fundada en 1955 por William Shockley en su ciudad natal de Palo Alto, California. Algunos empleados de

la compañía de Shockley se separaron para fundar Fairchild Semiconductor, y gente de Fairchild formó otras compañías, incluyendo Intel Corporation.

En 1964 IBM anunció una tercera generación de equipos de cómputo: Su familia Sistema 360 de macrocomputadoras. Cada uno de los procesadores de esta familia tenía un conjunto muy amplio de instrucciones internas que podía ejecutar, algunas de esas instrucciones eran especialmente útiles en aplicaciones científicas, mientras que otras eran más apropiadas para procesamiento de archivos; de este modo era posible utilizar la línea 360 de manera eficiente en los dos ambientes. De 1964 a la fecha se han introducido muchas otras familias mejoradas de procesadores. En forma colectiva, se podría considerar a estos procesadores como una "cuarta generación", pero la industria nunca se puso de acuerdo en la designación de generaciones en los años subsecuentes.

Cada generación sucesiva de hardware ha ido acompañada de reducciones substanciales en los costos, tamaño, emisión de calor y consumo de energía, y por incrementos notables en velocidad y capacidad.

De 1965 a 1974

El desarrollo de las microcomputadoras. Las computadoras construidas antes de 1965 eran macrocomputadoras, diseñadas para proporcionar en una localidad centralizada todo el poder de procesamiento que requería una organización. Este enfoque resolvió las necesidades de algunas organizaciones, pero había otras que no contaban con los fondos necesarios para adquirir sistemas grandes o que tenían aplicaciones especializadas que no podían ser procesadas en forma eficiente por una máquina grande centralizada.

Existía una necesidad obvia de computadoras mínimas de bajo costo para llenar los huecos que dejaba el enfoque de máquinas grandes, rápidas y centralizadas. Varios innovadores se dieron cuenta de esta necesidad y formaron compañías nuevas en la década de 1960, para producir estas máquinas mínimas. Los primeros procesadores en recibir el nombre de

minicomputadoras se desarrollaron y construyeron en 1965 por la Digital Equipment Corporation, DEC.

La introducción del tiempo compartido. En el ambiente de computación de principios de los setentas, los usuarios preparaban sus datos y sus programas y después los llevaban al centro de cómputo para ser procesados; el centro de cómputo reunía todas estas tareas y las introducía por lotes a la computadora en intervalos programados. El retraso inevitable que resulta de este procesamiento por lotes era muy frustrante para algunos usuarios. Para remediar esta situación John Kemeny y Thomas Kurtz, profesores de la Dartmouth College, decidieron llevar más lejos algunos conceptos de tiempo compartido que se había desarrollado en el Instituto Tecnológico de Massachussets (MIT). Tiempo compartido es un término que se emplea para describir un sistema de proceso que cuenta con varias estaciones independientes, de baja velocidad (relativamente), en línea y susceptibles, cada una de las estaciones proporciona al usuario acceso directo al procesador central. Kemeny y Kurtz desarrollaron programas especiales que permitían al procesador conmutar entre las distintas estaciones de estudiantes y llevar a cabo una parte del trabajo en el segmento de tiempo asignado a cada una, hasta finalizar el trabajo.

El microprocesador

A partir de 1965 cada año se duplicó el número promedio de componentes que se empacaban en una pastilla de silicio y este avance condujo a un suceso insólito: la creación de un microprocesador que podía colocarse en una sola pastilla. Un microprocesador contiene todos los circuitos necesarios para realizar funciones aritméticas lógicas y de control.

El origen del microprocesador se remonta a finales de la década de 1960, en esa época Victor Poor, ingeniero en electrónica de la Datapoint Corporation, estaba trabajando en el diseño y desarrollo de computadoras de aplicación especial; cada vez que se necesitaba un dispositivo diseñado a la medida, Poor y otros ingenieros iniciaban la labor de diseño desde cero, lo cual resultaba de alto costo por todos los recursos involucrados, de ahí que surge la necesidad de crear un microprocesador con los elementos básicos de

aritmética, lógica y de control de una computadora en una sola pastilla de silicio; la pastilla podría producirse en grandes cantidades y después programarse de maneras diferentes para realizar tareas especiales.

En 1969 Victor Poor y Harry Pyle, desarrollaron un modelo de pastilla microprocesadora, Poor llevó su modelo de "pastilla procesadora" a dos fabricantes de componentes, Texas Instrument e Intel Corporation, para su fabricación; estas reuniones no produjeron decisiones inmediatas, pero los fabricantes de componentes estaban en libertad de utilizar el concepto de pastilla microprocesadora y lo hicieron.

A finales de 1969, un ingeniero de Intell llamado Marcian "Ted" Hoff presentó sus ideas para el diseño de un microprocesador a los representantes de una compañía de calculadoras japonesa, en ese tiempo las calculadoras se construían a partir de pastillas de circuitos especializadas que podían realizar únicamente una función, pero la pastilla nueva de Hoff podía programarse para llevar a cabo varias funciones de cálculo especializadas. Se empezó a trabajar en él, el primer microprocesador, Intel 4004, podía ejecutar únicamente algunas instrucciones, y sólo podía manipular cantidades diminutas de datos en un momento dado; pero para el otoño de 1971 Intel había producido una pastilla más poderosa la 8008, y Texas Instrument entregaba ya un microprocesador. En 1974 Intel produjo un tercer microprocesador, el 8080, y quedaron establecidas las bases para el desarrollo de una computadora personal.

Desarrollo de las Computadoras Personales

Los pioneros de las computadoras personales en 1975 dos jóvenes programadores Bill Gates y Paul Allen estaban completando un programa que podía traducir instrucciones escritas en BASIC a los códigos de lenguaje de máquina que requerían los microprocesadores de Intel; se creó la Microsoft Corporation. Muchos de los competidores en la industria actual de las computadoras personales son firmas de varios miles de millones de dólares que cuentan con laboratorios de desarrollo bien equipados, en cambio, los pioneros de 1975 eran en su mayoría aficionados e ingenieros autodidactas a los que les fascinaba la naciente tecnología.

En la primavera de 1976, un joven técnico de la Hewlett-Packard llamado Steve Wozniak compró un microprocesador de MOS Technology y se propuso construir una computadora a partir de él. Esta computadora fue la Apple I, sólo se construyeron 200 Apple I, pero a finales del verano Wozniak ya estaba trabajando sobre el diseño de la Apple II.

A finales de 1977, las máquinas que dominaban el mercado eran la Apple II, el modelo TRS-80 de la Radio Shack Division de la Tandy Corporation y las docenas de otras marcas que estaban diseñadas para seguir los conceptos de interconexión eléctrica utilizados en la computadora Altair.

Años recientes

A finales de la década de 1970 las máquinas líderes en el ramo se estaban utilizando en todas partes: hogar, escuela y lugares de trabajo, además al comenzar la década de 1980, compañías como Atari y Commodore estaban produciendo sistemas de bajo costo para utilizarse en los hogares y se introducía una nueva generación de modelos de escritorio más poderosos para emplearse en las escuelas y oficinas. IBM entró en el mercado con su familia de computadoras PC logrando enorme éxito al estandarizar la industria del diseño de equipos de cómputo.

Hace menos de diez años se introdujeron productos de software nuevos que combinaban varias funciones dentro de un sólo paquete. El origen de estos paquetes integrados de software, que pueden mostrar la situación actualizada de varias aplicaciones en ventanas separadas dentro de la pantalla, se remota a principios de la década de 1970. Por esas fechas investigadores de Palo Alto Research Center de Xerox desarrollaron el primer software integrado con ventanas. Pero no fue hasta que Xerox introdujo su 8010 Star Information System, en 1981, cuando estas ideas aparecieron en un producto comercial. Los datos y programas se presentaban en la pantalla por medio de imágenes pequeñas llamadas íconos, los datos pueden pasarse de una aplicación a otra a través de las diferentes ventanas que pueden abrirse en la pantalla; dado que el Star costaba más de 15,000 dólares pocas personas se dieron cuenta de lo que podía hacer.

En Apple Computer, sin embargo, las ventajas del Star no pasaron inadvertidas influidos por los trabajos realizados en Xerox, los ingenieros de Apple diseñaron una computadora de menor costo que incluía muchas de las características de la Star. Este sistema de Hardware-Software, llamado Lisa, se anunció en 1983; en 1984 se introdujo la Apple Macintosh a un precio de 2,500 dólares y con muchas de esas características de hardware-software. Por supuesto, Apple no estaba sola, en 1984 se introdujeron docenas de paquetes integrados diseñados para las computadoras personales más populares y los vendedores de equipo se apresuraron a crear sistemas para apoyar la orientación gráfica que prefieren muchas personas al interactuar con computadoras personales.[SAN91]

SOFTWARE

Este consiste en los programas de instrucciones y datos que se definen para el Hardware y los algoritmos necesarios para la solución de problemas. La clasificación más conocida del software es la siguiente:

- 1. Aplicaciones**
- 2. Sistemas operativos**
- 3. Lenguajes de programación**

1. Aplicaciones

- Procesadores de texto
- Hojas de cálculo
- Manejadores de bases de datos
- Sistemas especializados

2. Sistemas operativos

Un sistema operativo es el conjunto de programas de control y programas de proceso que relacionados entre sí dan coherencia a las acciones que lleva a cabo una computadora optimizando el uso de la misma mediante un

aprovechamiento eficiente de todos los recursos y una ordenada comunicación entre usuario y computadora.

Los elementos de un sistema operativo son:

a) Programas de utilerías

- Programas organizadores del sistema
- Programas organizadores de archivos

b) Programas de servicio

- Traductores
 - + Compiladores
 - + Intérpretes
 - + Preprocesadores
 - + Ensambladores
- Programas de gestión de librerías
 - + Incorporación de programas
 - + Anulación de programas
 - + Organización de programas
 - + Funciones específicas de programas
- Editores

c) Programas de control

- Administradores de hardware
- Administradores de archivos
- Administradores de procesos

Los sistemas operativos al igual que el hardware de las computadoras, han sufrido una serie de cambios revolucionarios llamados generaciones.

Generación cero (década de 1940)

Los primeros sistemas computacionales no poseían sistemas operativos. Los usuarios tenían completo acceso al lenguaje de la máquina. Todas las instrucciones eran codificadas a mano.

Primera generación (década de 1950)

Los sistemas operativos de los 50's fueron diseñados para hacer más fluida la transición entre trabajos. Antes de que los sistemas fueran diseñados, se perdía un tiempo considerable entre la terminación de un trabajo y el inicio del siguiente. Este fue el comienzo de los sistemas de procesamiento por lote "batch", donde los trabajos se reunían por grupos o por lotes. Cuando el trabajo estaba en ejecución, este tenía control total de la máquina. Al terminar cada trabajo (exitosamente o anormalmente), el control era devuelto al sistema operativo, el cual "limpiaba" la máquina e iniciaba el trabajo siguiente.

Segunda generación (Primera mitad de la década de 1960)

La característica de la segunda generación de los sistemas operativos fue el desarrollo de los sistemas compartidos con multiprogramación, y los principios del multiprocesamiento. En los sistemas de multiprogramación, varios programas de usuarios se encuentran al mismo tiempo en almacenamiento principal, y el procesador se cambia rápidamente de un trabajo a otro. En los sistemas de multiprocesamiento se utilizan varios procesadores en un solo sistema computacional, con la finalidad de incrementar el poder de procesamiento de la máquina. La independencia de dispositivos aparece después. Un usuario que deseara escribir datos en una cinta en sistemas de la primera generación tenía que hacer referencia específica a una unidad de cinta particular. En los sistemas de la segunda generación, el programa del usuario especificaba tan sólo que un archivo iba a ser escrito en una unidad de cinta con cierto número de pistas y cierta densidad. El mismo sistema operativo localizaba entonces, una unidad de cinta disponible con las características deseadas, y le indicaba al operador que montara una cinta en esa unidad. Se desarrollaron sistemas de tiempo compartido, en los que los usuarios podían acoplarse directamente con la

computadora a través de terminales parecidas a máquinas de escribir. Los sistemas de tiempo compartido operan en modo interactivo o conversacional con los usuarios. El usuario teclea una petición a la computadora, ésta la procesa tan pronto le sea posible (generalmente en el transcurso de un segundo o menos), y la respuesta (si la hay), aparece desplegada en la terminal del usuario. La computación conversacional hizo posible grandes adelantos en el proceso de desarrollo de programas. Un usuario de tiempo compartido podía, en segundos o minutos, localizar y corregir errores; mientras que, en un ambiente de procesamiento por lotes, el usuario tenía que sufrir grandes retrasos, a menudo de horas o días. Surgieron sistemas de tiempo real, en que las computadoras fueron utilizadas en el control de procesos industriales, como en la refinación de gasolina. Los sistemas militares de tiempo real fueron desarrollados para regular o supervisar, en caso de sufrir un ataque aéreo, miles de puntos al mismo tiempo. Los sistemas de tiempo real se caracterizan por proveer una respuesta inmediata. Por ejemplo, una lectura obtenida en una refinería de gasolina, indicando que las temperaturas están muy altas, puede demandar atención inmediata para poder evitar una explosión. Este hecho ayuda a explicar su alto costo.

Tercera generación(segunda mitad de la década de 1960 a la primera mitad de la década de 1970)

La tercera generación de sistemas operativos comenzó en forma efectiva. En 1964, con la introducción de la familia de computadoras sistema 360 de IBM. Las computadoras de la tercera generación fueron diseñadas como sistemas de usos generales. Casi siempre eran sistemas grandes, voluminosos. Este concepto vendió una gran cantidad de computadoras, pero tenía su precio. Los usuarios que necesitaban de algunas aplicaciones particulares que no requerían de este tipo de poder pagaban caro el elevado tiempo de ejecución, tiempo de aprendizaje, tiempo de depuración, mantenimiento, etc. Los sistemas de la tercera generación eran sistemas de modos múltiples. Algunos de ellos soportan simultáneamente procesos por lotes, tiempo compartido, procesamientos de tiempo real y multiprocesamiento. Eran grandes y costosos, nunca antes se había construido algo similar, y muchos de los esfuerzos de desarrollo terminaron muy por arriba del presupuesto y mucho después de que lo que el planificador marcaba como fecha de terminación

(una notable excepción a esto es el sistema UNIX, desarrollado en los laboratorios Bell). Estos sistemas introdujeron mayor complejidad a los ambientes computacionales; una complejidad a la cual, en un principio, no estaban acostumbrados los usuarios. Los sistemas interponían una capa de software entre el usuario y el hardware. Esta capa de software era a veces tan gruesa que el usuario perdía de vista al hardware, y veía sólo el punto de vista creado por el software. Para lograr que uno de estos sistemas realizara la tarea más simple, los usuarios debían familiarizarse con un complicado lenguaje de control de trabajos, a fin de poder especificar el trabajo y los recursos requeridos. Los sistemas operativos de la tercera generación representaron un gran paso hacia adelante, pero también doloroso para muchos usuarios.

Cuarta generación (segunda mitad de la década de 1970 a nuestros días)

Los sistemas operativos de la cuarta generación constituyen el estado actual de la tecnología. Muchos diseñadores y usuarios se sienten aún incómodos, después de sus experiencias con los sistemas operativos de la tercera generación y se muestran cautelosos antes de comprometerse con sistemas operativos complejos. Con la ampliación del uso de las redes de computadoras y del procesamiento en línea, los usuarios obtienen acceso a computadoras alejadas geográficamente a través de varios tipos de terminales. El microprocesador ha hecho posible la aparición de la computadora personal, uno de los desarrollos de notables consecuencias sociales de las últimas décadas. Ahora, muchos usuarios han desarrollado sistemas de computación que son accesibles para su uso personal en cualquier momento del día o de la noche. La potencia del computador, que costaba varios cientos de miles de dólares al principio de la década de 1960, hoy es accesible por menos de mil dólares. Por lo general las computadoras personales están equipadas con interfaz para comunicación de datos, y también sirven como terminales o también pueden funcionar como estaciones de trabajo. El usuario de un sistema de la cuarta generación ya no se encuentra confinado a la comunicación con una sola computadora en un modo de tiempo compartido. En lugar de esto, el usuario puede comunicarse con sistemas alejados geográficamente. Los problemas de seguridad se han incrementado mucho ahora que la información pasa a

través de varios tipos vulnerables de líneas de comunicación. La información cifrada en claves recibe mucha atención ; ha sido necesario codificar los datos personales, confidenciales, para que aun si los datos son expuestos, no sean de utilidad a nadie más que a los receptores adecuados. El porcentaje de la población que tiene acceso a una computadora en las décadas de los ochenta y noventas se ha incrementado mucho más que nunca y se pronostica que aumentará, es común escuchar el término favorable al usuario, para referirse a sistemas que permiten un fácil acceso al poder computacional, a usuarios de inteligencia normal. Los ambientes del usuario, altamente simbólicos, mnemónicos y orientados de los 60's y 70's, son reemplazados en la época de los 80's y 90's, por los sistemas controlados por menú, los cuales guían al usuario a lo largo de varias opciones expresadas en un lenguaje sencillo. El concepto de máquinas virtuales es muy utilizado. El usuario ya no se encuentra interesado en los detalles físicos del sistema de computación o de la red, que esta siendo accedida. En su lugar , el usuario ve un panorama llamado máquina virtual creado por el sistema operativo. El usuario actual se interesa más por el cumplimiento de un trabajo con una computadora, y en general no se preocupa por el funcionamiento interno de la máquina. [DEI89]

3. Lenguajes de programación

Los lenguajes de programación son el medio de comunicación entre el programador y la computadora, con ellos se construyen los programas que serán ejecutados por las computadoras.

Los lenguajes se clasifican en dos categorías:

- Lenguajes de bajo nivel.
- Lenguajes de alto nivel.

Los lenguajes de bajo nivel son los lenguajes ensamblador y máquina. En estos lenguajes cada línea del programa constituye una instrucción específica para la computadora; en el caso de los lenguajes ensamblador cada instrucción posee un mnemónico. En el lenguaje máquina, las instrucciones se expresan en código binario o hexadecimal por lo que la programación en estos lenguajes resulta compleja.

Tanto los programas escritos en lenguaje ensamblador como los escritos en lenguaje de alto nivel son traducidos a lenguaje máquina antes de ser ejecutados, ya que es este el único comprensible por la computadora. Los programas en lenguaje ensamblador, son traducidos mediante un programa llamado ensamblador, y los de alto nivel se traducen a través de compiladores e intérpretes.

Cada instrucción creada con un lenguaje de alto nivel se traduce en varias ordenes para la computadora. Al compilarse un programa escrito en lenguaje de alto nivel, cada instrucción es implementada mediante varias instrucciones en lenguaje máquina, creándose un programa en lenguaje máquina que realiza el proceso descrito en lenguaje de alto nivel.

Al programar en lenguaje ensamblador se obtienen programas mucho más eficientes y rápidos, ya que se realizan consideraciones sobre la arquitectura de la computadora. Los programas escritos en lenguajes de alto nivel nos permiten realizar código transportable, es decir, elaborar programas que puedan ser ejecutados en diferentes computadoras con un mínimo de modificaciones requeridas.

Las primeras computadoras fueron desarrolladas en el período comprendido entre 1937 y 1945. Toda la programación se llevaba a cabo en lenguaje máquina. El concepto de almacenamiento y ejecución automática de un programa fue sugerido por primera vez por John Von Newman en 1946. Entre 1946 y 1955, los cambios en el hardware fueron tantos y tan rápidos, que una computadora se volvía obsoleta antes de que cualquier optimización importante de software fuera intentada. En 1955 la computadora IBM 650 se convirtió en el primer éxito comercial; esta computadora se programaba en SOAP (Symbolic Optimizer and Assembly Program). El primer lenguaje de alto nivel para computadoras comerciales fue FORTRAN (Formula Translation Language), fue desarrollado para la computadora IBM 704 entre 1957 y 1959. Durante este mismo periodo, el lenguaje Algol (Algorithmically Oriented Language) fue desarrollado por la unión de la ACM (American Association for Computer Machinery), y la Asociación Profesional Europea de Especialistas en Computación, GAMM. A partir de este momento se crearon

un gran número de lenguajes como COBOL (1959 -1960), PL/1(1963 -1964), BASIC (1963), APL (1966), PASCAL (1971 -1973), etc.

En la actualidad existen más de cien lenguajes de programación, entre los cuales se encuentra el Visual BASIC que es una variante del lenguaje BASIC, que proporciona herramientas para programación en ambiente windows.

El nombre de BASIC responde a las siglas de Begginer's All-purpose Symbolic Instruction Code. Este lenguaje fue desarrollado, junto con el primer sistema de tiempo compartido del mundo. Sus autores fueron John G. Kemeny y Thomas E. Kurts. BASIC es un intérprete, es decir, cada instrucción se traduce a lenguaje máquina e inmediatamente después es ejecutada.

Visual Basic soporta además del "set" de instrucciones del BASIC estándar, un conjunto de objetos que pueden ser utilizados en cualquier aplicación desarrollada en este lenguaje, estos objetos pueden ser manipulados a través de un conjunto de propiedades y métodos, y la programación se basa en la respuesta de dichos objetos a los diferentes eventos ocurridos en el ambiente. Como puede observarse este lenguaje de programación ha sido una adaptación de BASIC a la filosofía de programación con objetos, dirigida por eventos.

RECURSOS HUMANOS

En este elemento se debe poner la mayor atención, porque de este factor dependerá por quién estará hecho el sistema, por quién será probado y para quién estará hecho.

- El usuario
- El analista
- El Programador
- El líder del proyecto
- El Administrador del sistema
- Los derechos

Usuario

Un sistema basado en computadora casi siempre tiene un elemento humano. Una persona puede interactuar directa o indirectamente con el hardware y el software, manteniendo un diálogo que conduzca las funciones del sistema; en todos los casos la gente es responsable del desarrollo, soporte, administración y mantenimiento del sistema.

El usuario es la persona quien se beneficia directa o indirectamente de las capacidades de los sistemas de cómputo y utiliza esas capacidades para agilizar o mejorar sus actividades profesionales, técnicas o laborales.

p.ej. Llevar a cabo un análisis financiero de una compañía, publicar un reporte de calidad, mantenimiento del inventario, etc.

Con la incorporación del procesamiento de datos durante las décadas de los 50's y 60's, el acceso a los sistemas de procesamiento de datos estaba muy restringido y muy poca gente tenía contacto directo con ellos.

Actualmente con la distribución y difusión de herramientas de software y de equipos de cómputo el número de usuarios se ha incrementado de manera muy significativa y existe una gran variedad de ellos. Además otro aspecto importante es el gran decremento del costo de los equipos de cómputo llegando al grado de que se puede considerar a un equipo de cómputo como equipo de oficina, herramienta didáctica e incluso como un aparato de tipo doméstico.

Entorno del usuario

El hardware, el software y otros elementos del sistema se combinan para formar un entorno del usuario. El entorno puede incluir facilidades físicas, así como la propia interacción hombre-máquina.

Analista

El analista debe establecer contacto con el equipo técnico de gestión del usuario/cliente y con la empresa que vaya a desarrollar el software. El

objetivo del analista es reconocer los elementos básicos del programa tal como lo percibe el usuario/cliente.

Se espera que el analista de sistemas analice y diseñe sistemas con un rendimiento óptimo; esto es, el analista debe producir una salida que cumpla con los objetivos de gestión. El analista debe exhibir los siguientes rasgos de carácter :

Habilidad para comprender conceptos abstractos, reorganizarlos en divisiones lógicas y sintetizar " soluciones " basadas en cada división.

Habilidad para entresacar hechos importantes de fuentes conflictivas o confusas.

Habilidad para comprender entornos de usuario/cliente.

Habilidad para aplicar elementos hardware y/o software a entornos de usuario/cliente.

Habilidad para comunicarse bien en forma escrita y verbal.

Habilidad para concretarse a los objetivos globales de los programas.

Los requerimientos del software deben ser descubiertos de una manera descendente; las funciones importantes, interfaces e información deben comprenderse completamente antes de que se especifiquen los detalles de las etapas decisivas.

El analista ejecuta o coordina cada una de las tareas asociadas con el análisis de los requerimientos de software. Durante las tareas de reconocimiento, se comunica con el equipo del usuario/cliente para establecer las características del entorno existente; luego el analista contacta con el equipo de desarrollo durante las tareas de evaluación y síntesis, de forma que se definan correctamente las características del software. Generalmente el analista es el responsable del desarrollo de una especificación de los requerimientos de software y también participa en todas las revisiones.

Es importante observar que el analista debe comprender también cada uno de los paradigmas y apreciar las fases y pasos genéricos de la ingeniería de la programación que se aplican independientemente de la metodología usada. Muchos requerimientos implícitos se incorporan en la especificación de los

requerimientos, sólo y sólo si el analista comprende la ingeniería del software.

Programador

La programación real se lleva a cabo por personal que se especializa en el área de desarrollo de sistemas. Con más frecuencia se les llama programadores de aplicaciones. Muchos analistas comienzan como programadores y después se convierten en analistas de sistemas. ¿Los analistas de sistemas escriben programas para computadoras ? En realidad, es parte del perfil de algunas personas conocidas a menudo como analistas programadores. ¿Realizan programas para computadoras la mayoría de los analistas en la actualidad ? Esto realmente varía; sin embargo, una cosa es muy clara: los mejores y más valiosos analistas de sistemas saben programar. Los analistas que tienen este tipo de conocimientos normalmente son más valiosos, ya que su experiencia adicional les permite formular mejores y más completas especificaciones para una nueva aplicación. No solamente conocen lo que puede hacerse o no dentro de un programa, sino que saben como hacerlo y también qué es lo que debe de comunicar el programador. El resultado es casi siempre un software de mayor calidad y con un tiempo menor de desarrollo.

Líder de proyecto

El líder de proyecto tiene la responsabilidad de realizar los estudios completos de los sistemas, así como también la responsabilidad de diseñarlos. Algunos atributos que un buen líder de proyecto debe de poseer para lograr un mejor desempeño son los siguientes:

- Aprender a aceptar críticas
- Tomar riesgos
- Delegar responsabilidades
- Ser persistente
- Pensar antes de actuar
- Asesorar al equipo de trabajo
- Mantener actitud perseverante

- Enfrentar compromisos

Administrador del sistema

El Administrador es quien está a cargo del control del sistema en todo lo referente al entorno del mismo. Con esto nos referimos a que el administrador selecciona al personal que puede utilizar el sistema de manera directa o indirecta. También tiene la responsabilidad de organización para la efectividad de los sistemas.

Derechos

El Diseño y desarrollo del sistema " Tutor " no está planeado para fines de comercialización, por lo que el uso de bibliografías para fines de apoyo en nuestras áreas de evaluación del sistema no cae en alguna irregularidad referida a violaciones de derechos de autor. En el caso de que se proyectara comercializar el sistema, se tendría que proceder de la manera adecuada para evitar caer fuera de lo permitido; es decir, entablar los acuerdos necesarios para poder lanzar al sistema tutorial como un producto académico de apoyo a estudiantes con aspiraciones al ciclo de bachillerato.

BASES DE DATOS

En el contexto más amplio de la definición de base de datos no nos referimos solamente al manejador de bases de datos y a los archivos que contienen la información correspondiente a los exámenes, si no a estos y a todas las partes involucradas con la información correspondiente a la elaboración del examen de admisión, para el CCH y la Escuela Nacional Preparatoria e instituciones privadas, esto quiere decir que tanto los criterios de calificación de los exámenes, la calidad, precisión y actualidad con que se trata cada una de las diez materias propuestas por la guía de estudios, el seguimiento de una metodología y la selección más adecuada de todos los elementos de que se compone el sistema estarán presentes en la información proporcionada al alumno.

La organización en una base de datos debe representar el significado de fondo o semántica de los datos en forma correcta y eficiente . En los programas convencionales, la estructura de los datos se arregla de acuerdo con la conveniencia del programa. Una base de datos contiene datos que serán utilizados por muchos y diferentes programas. Debido a esto, la organización de una base de datos no puede estar exclusivamente determinada por decisiones tomadas al programar funciones específicas. Los archivos son las unidades físicas principales en las que se organiza una base de datos. El almacenamiento de datos para una base de datos se logra empleando uno o más archivos. Un archivo se define como un conjunto de registros semejantes conservados en dispositivos de almacenamiento secundario en una computadora. Los dispositivos de almacenamiento secundario son discos magnéticos, aunque existen ciertas alternativas que no utilizaremos. Un registro es un conjunto de campos relacionados que contienen datos. Un dato está representado por un valor que es parte de una descripción de un objeto o evento. Los procesos de cómputo pueden manejar tales valores. Las medidas referentes al tamaño de los archivos dependen directamente del hardware y de las restricciones operativas aplicables en un ambiente dado; los archivos no solamente se caracterizan por su tamaño, sino que se distinguen aún más de acuerdo con su organización. Las diferencias en la organización de archivos llevan a grandes diferencias en el desempeño al almacenar y recuperar archivos (escritura/lectura). A menudo una base de datos requiere más de un tipo de archivo. Al leer o escribir archivos, los datos se transfieren entre los dispositivos de almacenamiento y de memoria del sistema de computación, por lo que se define la entrada/salida con los términos lectura/escritura correspondientemente y es el sistema operativo el que proporciona capacidades adecuadas de entrada y salida. Los procesos son las unidades básicas de computación manejadas por un sistema operativo.

DOCUMENTACIÓN

Dependerá básicamente del modelo conceptual utilizado y de la metodología con que se desarrolló el sistema; en seguida listamos los documentos referentes a este punto:

- **Introducción**
- **Objetivos del proyecto**
- **Reporte de análisis**
- **Especificación del sistema**
- **Documentación de la arquitectura**
- **Diseño del programa de computadora**
- **Código**
- **Planes de prueba**
- **Procedimientos de prueba**
- **Manual de usuario**
- **Manual técnico**

II. ANÁLISIS

II.1 DOCUMENTO DE OBJETIVOS DEL PROYECTO

En este documento se resumen los resultados de la etapa conocida como planteamiento de objetivos, en la cual se identifican y describen las necesidades del usuario, a fin de proponer un conjunto de objetivos que de lograrse implicarían la satisfacción de dichas necesidades [IIE85].

IDENTIFICACIÓN Y DESCRIPCIÓN DE LAS NECESIDADES DEL USUARIO

Actualmente los aspirantes a nivel medio superior necesitan de apoyo académico para cumplir con el objetivo de ser aceptados en el plantel de su preferencia.

Se requiere de métodos atractivos que induzcan al aspirante a una preparación autodidacta que le permita alcanzar el nivel académico exigido, ya sea mediante el estudio secuencial y sistemático de los temas, o bien concentrando su atención en aquellas áreas en las que considere debe mejorar su preparación.

El alumno debe contar con la seguridad suficiente para la presentación de su examen, por lo que le es preciso conocer sus avances.

El alumno debe alcanzar un buen nivel académico al llegar a la etapa de educación media superior.

OBJETIVOS DEL PROYECTO

Desarrollar un sistema de computadora orientado a la preparación de aspirantes a escuelas de nivel medio superior.

Elaborar una nueva herramienta que sirva como una alternativa para que el aspirante pueda complementar su instrucción.

La creación de un sistema que emplee la computadora como un auxiliar en la educación, que por sus características innovadoras y a través de una interface interactiva resulte en sí una motivación para que el aspirante dedique tiempo a su preparación.

El proporcionar un medio de estudio flexible, que permita al aspirante decidir el tipo de preparación que más se adapte a sus necesidades, ya sea cubriendo ordenadamente un examen global, o atacando directamente aspectos deficientes de su preparación en una materia específica.

Dar al aspirante elementos para que pueda evaluar su rendimiento, determinando deficiencias o avances en un área específica o a nivel general y comparar su desempeño con el nivel exigido por las escuelas de nivel medio superior.

II.2 REPORTE DE ANÁLISIS

Este documento es el resultado de la etapa de análisis, durante la cual se deben estudiar los objetivos para establecer un compromiso entre los objetivos conflictivos o antagónicos, señalar prioridades, constatar factibilidad y proponer los métodos de solución que sirvan de base para el diseño del sistema.[IIE85]

JUSTIFICACIÓN

Se justifica el desarrollo del sistema " Tutor " para aspirantes al examen de admisión al ciclo de bachillerato (CCH's.,E.N.P.,etc.) debido a la necesidad de crear alternativas para que el aspirante complemente su instrucción. Siendo una motivación para el estudio el ambiente interactivo, el cual proporciona elementos para la autoevaluación del rendimiento. Permitiendo determinar avance o deficiencia en un área específica respecto al nivel exigido por la gafa de estudios para realizar el mencionado examen.

ALCANCES

Los alcances que tendrá el sistema " Tutor " abarcan las siguientes áreas de estudio: Español, Literatura, Historia Universal, Historia de México, Geografía General, Geografía de México, Matemáticas, Física, Química y Biología. A continuación se dan los temarios correspondientes:

ESPAÑOL

1. Generalidades Lingüísticas
2. Morfología
3. Semántica
4. Gramática o morfosintaxis
5. Expresión escrita
6. Lectura de comprensión

LITERATURA

1. Generalidades
2. Literaturas antiguas
3. Literaturas clásicas
4. Literaturas medievales europeas
5. Literatura española del renacimiento
6. Literaturas prehispánicas
7. Literatura barroca
8. Literatura neoclásica
9. Literatura romántica
10. Literatura realista
11. Literatura modernista

HISTORIA UNIVERSAL

1. Estudio de la Historia
2. Principales civilizaciones de la antigüedad
3. El Mundo Medieval
4. El Mundo Moderno
5. El Mundo Contemporáneo
6. El Mundo Actual

HISTORIA DE MÉXICO

1. México prehispánico
2. La conquista de México
3. México Colonial
4. La Guerra de Independencia
5. México Independiente
6. El Porfiriato y la Revolución Mexicana
7. México Actual

GEOGRAFIA GENERAL

1. Medio ambiente
2. La Tierra
3. Litosfera
4. Hidrosfera
5. Atmósfera
6. América
7. Europa
8. África
9. Asia
10. Sistemas económico-políticos

GEOGRAFÍA DE MÉXICO

- 1. Situación de México**
- 2. Características físicas de México**
- 3. Población**
- 4. Zonas económicas**

MATEMÁTICAS

- 1. Números naturales**
- 2. Números enteros**
- 3. Números racionales**
- 4. Álgebra**
- 5. Geometría**
- 6. Trigonometría**

FÍSICA

- 1. Unidades de Medición**
- 2. Mecánica**
- 3. Ondas**
- 4. Propiedades y estados de la materia**
- 5. Termodinámica**
- 6. Electricidad**
- 7. Electromagnetismo**

QUÍMICA

- 1. Introducción al estudio de la química**
- 2. Modelos y características del átomo**
- 3. Clasificación de los elementos**
- 4. Enlaces**
- 5. Ácidos y bases**
- 6. Cálculos Químicos**
- 7. Reacciones Químicas**

BIOLOGÍA

1. El Estudio de la Biología
2. Niveles de organización biológica
3. Los seres vivos y la energía
4. Madurez sexual y reproducción humana
5. Genética
6. Salud
7. Evolución y sus características
8. Condiciones físico-químicas del medio ambiente
9. Comunidades bióticas

[SCH92]

Dentro del sistema " Tutor " se generan dos tipos de exámenes:

a) Examen por Sección.

Existen dos variantes para el examen por sección: (1) Modo Tutor, (2) Modo examen.

El examen por sección en "Modo Tutor" consistirá de un bloque de treinta preguntas de tipo aleatorio tomadas de las bases de datos de acuerdo a la materia que sea seleccionada; cada examen proporcionará un límite de tiempo adecuado a la materia examinada, al término del examen (ya sea por tiempo o por haber concluido el mismo) se podrán evaluar los resultados obtenidos. Durante este tipo de examen el usuario tendrá acceso al sistema de ayuda.

El examen por sección en "Modo examen" (valga la redundancia) consistirá en un conjunto de treinta preguntas concernientes a la materia seleccionada, este examen a diferencia del tipo "Tutor" no se calificará inmediatamente cada una de las preguntas, sino hasta que se finalice éste. Otra diferencia respecto al examen de tipo "Tutor" es que no contará con el sistema de ayuda (siendo similar a un examen de tipo real).

b) Examen Global.

Este examen consistirá de un bloque de trescientas preguntas de tipo aleatorio tomadas de las diez diferentes bases de datos, siguiendo el siguiente criterio:

Materia	Número de Preguntas
Biología	35
Español	25
Física	40
Geografía General	25
Geografía de México	25
Historia de México	25
Historia Universal	25
Literatura	25
Matemáticas	40
Química	35
TOTAL	300

El orden con que serán presentados los diferentes temas será aleatorio. Cada examen constará de tres horas y media como límite de tiempo para su realización.

Además el sistema será capaz de calificar exámenes tanto de tipo global como por sección y generar reportes de calificaciones.

LIMITACIONES

Las Limitaciones del sistema están enfocadas hacia dos puntos básicos. Por un lado, el uso de esta herramienta estaría reservada únicamente para quienes tienen acceso a una computadora personal con las características requeridas por el sistema " Tutor ". Por otra parte, el sistema abarca ciertas áreas de estudio; es decir, que en el momento en que varíen los temarios considerados en los exámenes de admisión, el sistema " Tutor " quedaría

desactualizado y habría que hacer un reanálisis de los temarios que éste considera.

PRUEBA DE LA FACTIBILIDAD DEL PROYECTO

La recolección de datos que se lleva a cabo durante la investigación preliminar examina la factibilidad del proyecto; es decir, la posibilidad de que el sistema sea benéfico para el usuario. Se estudian tres pruebas de factibilidad: operativa, técnica y financiera.

Factibilidad operativa

Los proyectos propuestos son benéficos sólo si pueden convertirse en sistemas de información que cumplan con los requerimientos operativos. Dicho sencillamente, esta prueba de factibilidad cuestiona si el sistema trabajará cuando se desarrolle e instale.

Al introducir un sistema " Tutor " con perspectivas totalmente diferentes a los métodos comúnmente utilizados, ayuda a la automotivación de los aspirantes a través de la obtención de resultados positivos. Por este camino puede lograrse un apoyo total de todos los usuarios que tengan la posibilidad de adquirir el software.

El recurrir a cursos de regularización es un medio limitado a un cierto número de horas y de exámenes tipo; en cambio, el sistema " Tutor " no se limita en este aspecto, por el contrario, obliga a la consulta de bibliografías para obtener los conocimientos necesarios que alcancen el nivel de las evaluaciones del tutorial. Esto trae como resultado un mayor tiempo invertido en autoestudio y autoevaluación que no limita la preparación de los aspirantes.

El sistema " Tutor " amplía la fuente de conocimientos referidos a los temarios, debido al tipo de exámenes aleatorios que siempre proporciona. En todo momento, contribuye a un estudio más concientizado y detallado que sin duda conduce a un mejor desempeño individual.

Factibilidad técnica

Los aspectos técnicos que normalmente surgen durante la etapa de factibilidad son:

Tecnología necesaria para desarrollar el sistema " Tutor "

Configuración mínima recomendable

- 80 MB en disco duro
- 2 MB de memoria RAM
- FD 1.4 MB 3.5 "
- Procesador 80-386 a 16 MHz

Los requerimientos en Software son:

- Sistema operativo "DOS" en cualquiera de sus variantes (Ver 5.0 o superior) y el ambiente operativo MS-Windows 3.1
- Visual BASIC versión 3.0 edición profesional
- DBase IV
- Word para Windows en cualquiera de sus versiones
- Power Point 2.0

Factibilidad financiera y económica

Un sistema que puede desarrollarse técnicamente y que se utilizará si se instala, debe de considerarse como una buena inversión; es decir, los beneficios financieros deben igualar o exceder los costos financieros.

El costo estimado de hardware, software, investigación y desarrollo del sistema fue el siguiente:

Nota: Los costos del hardware y software utilizados se cotizaron en dólares al tipo de cambio actual y no en pesos debido a que es un estándar dentro del mercado computacional .

Para determinar el costo por la utilización del equipo para el desarrollo del sistema " Tutor " consideramos lo siguiente:

La Ley de Impuesto sobre la Renta (LISR), en la parte relativa a la depreciación de equipo de cómputo.

Como sabemos, la depreciación es la baja de valor de un bien tangible y la amortización es la baja de valor de un bien intangible, en ambos casos por el uso, el transcurso del tiempo o la obsolescencia. La LISR establece ciertas tasas de depreciación y amortización aplicables fiscalmente en cierto número de años, que no siempre tienen que ver con la verdadera vida estimada del bien.

Transcribiremos unos cuantos artículos de la LISR para posteriormente comentarlos.

" Art. 42.- Concepto de inversiones. Para efectos de esta Ley se consideran inversiones los activos fijos, los gastos diferidos, cargos diferidos y las erogaciones realizadas en periodos preoperatorios, cuyo concepto se señala a continuación:

Activo Fijo. Es el conjunto de bienes tangibles que utilizan los contribuyentes para la realización de sus actividades y que se demeriten por el uso en el servicio del contribuyente y por el transcurso del tiempo. La adquisición o fabricación de estos bienes tendrá siempre como finalidad la utilización de los mismos, para el desarrollo de las actividades del contribuyente y no la de ser enagenados dentro del curso normal de sus operaciones.

Gastos diferidos. Son los activos intangibles representados por bienes o derechos que permitan reducir costos de operación o mejorar la calidad o aceptación de un producto, por un período determinado, inferior a la duración de la actividad de la persona moral.

Cargos diferidos. Son aquellos que reúnen los requisitos señalados en el párrafo anterior, pero cuyo beneficio sea por un periodo ilimitado que dependerá de la duración de la persona moral.

Artículo 43.- Los porcentos máximos utilizados tratándose de gastos y cargos diferidos son los siguientes:

- I. 5% para cargos diferidos.
- II. 10% para otros gastos diferidos.

Artículo 44.- Los porcentos máximos autorizados, tratándose de activos fijos por tipo de bien son los siguientes: (las fracciones de la I a la VI no se transcriben por que no se aplican a equipo de cómputo).

VII. Tratándose de equipo electrónico:

a) 25% para equipo consistente de una máquina o grupo de máquinas interconectadas conteniendo unidades de entrada, almacenamiento, computación, control y unidades de salida, usando circuitos electrónicos en los elementos principales, para ejecutar operaciones aritméticas o lógicas en forma automática por medio de instrucciones programadas, almacenadas o controladas externamente.

b) 12% para equipo *periférico* del contenido en el inciso anterior de esta fracción; perforadoras de tarjetas, verificadoras, tabuladoras, clasificadoras, intercaladoras y demás que no queden comprendidas en dicho inciso.

IX. 100% para máquinas registradoras de comprobación fiscal."

Vale la pena mencionar que existe una opción de "deducción inmediata de inversiones" (artículo 51), mediante la cual se puede aplicar un 89% para equipo de cómputo y un 81% para equipo periférico en un lapso de un año. El resto, para llegar al 100%, no se puede deducir nunca. Esto no es aplicable en el D.F., Guadalajara y Monterrey, ni en sus áreas metropolitanas. Todo lo antes transcrito de la ley es para empresas. Para personas físicas se autoriza también para equipo de cómputo el 25% y el 12% para equipo periférico. La ley no menciona los cargos y gastos diferidos. En este inciso b) que dice, "12% para equipo *periférico* del contenido en el inciso anterior de esta fracción", y nunca define que es equipo periférico, por lo que es importante mencionar que nadie considera como equipo periférico para efectos contables o fiscales, el teclado, monitor o las unidades de discos debido a que si se elimina cualquiera de estos no opera el equipo de cómputo(en nuestro caso PC's). [PCM94]

Hardware

Características del hardware utilizado para la programación del sistema.

- 200 MB en disco duro 3472 dils
- 8 MB de memoria RAM
- FD 1.2 MB 5.1/4"
- FD 1.4 MB 3.5 "
- Procesador pentium a 60 MHz

Costo de utilización: $\left(\frac{3472}{48}\right)^{3.5} \approx 254$

Se usaron para la captura de las bases de datos de las áreas de estudio consideradas por el sistema " Tutor ", y para la edición de textos relativos al sistema.

- 40 MB en disco duro 400 dils
 - 1 MB de memoria RAM
 - FD 1.4 MB 3.5 "
 - Procesador 80-286 a 12 MHz
-
- 200 MB en disco duro 1479 dils
 - 8 MB de memoria RAM
 - FD 1.2 MB 5.1/4
 - Procesador 80386 a 25 MHz
-
- 200 MB en disco duro 1529 dils
 - 4 MB de memoria RAM
 - FD 1.2 MB 5.1/4
 - FD 1.4 MB 3.5 "
 - Procesador 80386 a 20 MHz

Costo de utilización: $\left(\frac{3408}{48}\right)^{1.5} \approx 107$

El siguiente equipo se usó para cargar las bases de datos de las diferentes áreas de estudio, además de la edición de los archivos que se utilizarán en los menús de ayuda y para la edición de preguntas que requieren objetos de tipo "fórmula" y elementos gráficos en power point.

- 100 MB en disco duro 1443 dlla
- 8 MB en RAM
- FD 1.2 MB 5.1/4"
- Procesador 80-386 a 25 MHz

$$\text{Costo de utilización: } \left(\frac{1443}{48} \right)^2 \approx 60$$

El siguiente equipo se uso para cargar las bases de datos de las diferentes áreas de estudio, además de la edición de los archivos que se utilizaron en los menús de ayuda

- 100 MB en disco duro 1324 dlla
- 2 MB en RAM
- FD 1.2 MB 5.1/4"
- Procesador 80-386 a 25 MHz

- 130 MB en disco duro 1384 dlla
- 6 MB en RAM
- FD 1.2 MB 3.1/2
- Procesador 80-386 a 25 MHz

$$\text{Costo de utilización: } \left(\frac{2708}{48} \right)^3 \approx 169$$

Periféricos.

El equipo periférico que se utilizó es el siguiente:

•Impresora laser HP laserjet III	1010 dlls
•Impresora Sharp JX-9500H emuladora HP laserjet IID	1145 dlls
•Impresora Star XR-1520 emuladora Epson	598 dlls
•Scanner HP color +	1440 dlls

Costo de utilización: $\left(\frac{4193}{100}\right)^4 \approx 168$

Software

WINDOWS	150 dlls
OFFICE	750 dlls
POWER POINT	
WORD FOR WINDOWS VER. 2.0	
VISUAL BASIC PROFESSIONAL 3.0	200 dlls
DBASE IV (DOS)	495 dlls

El costo del software se incluye totalmente en el costo del proyecto, por la naturaleza de las licencias, ya que se adquiere para ser entregado con el sistema (archivos ejecutables, librerías, paquetería necesaria), ya que en este caso no es posible entregar únicamente los ejecutables, y siempre es aconsejable que el usuario final cuente con el software de desarrollo para hacer factible el mantenimiento correctivo y adaptativo.

El costo total del hardware	: \$ 590.00 dlls
El costo total en periféricos	: \$ 168.00 dlls
El costo total del software	: \$ 1, 595.00 dlls

Horas Programador/Analista.

Para este punto obtuvimos un costo de \$10.00 dlls por hora, basados en el promedio de percepciones mensuales actuales de cada uno de los integrantes.

Se trabajó un total de 1,300 horas en el transcurso de tres meses y medio.

Costo de programación : \$ 13, 000.00 dlls

Concepto	Costo dlls
Hardware	590.00
Periféricos	168.00
Software	1, 595.00
Horas / Programador	13,000.00
Otros	2,750.00
Total	18,103.00

Considerando que el precio actual de la guía de estudios es de N\$ 40.00 precio al público y el costo de los tres discos de instalación requeridos por el sistema, no rebasa los N\$ 10.00. Determinando un costo del sistema "Tutor" de N\$ 50.00, se tendrían que vender 1,262 "paquetes" para abatir costos de desarrollo; tomando en cuenta que el tiraje anual de la guía de estudios es de 20,000 ejemplares, la cantidad (1,262) para abatir el costo representa tan sólo un 6.31 %. De acuerdo a lo mencionado anteriormente concluimos que : El sistema es factible.

De llevarse a cabo un analisis más profundo, deberían considerarse otros gastos tales como: publicidad, estudios de mercado, gastos de distribución, etc.

Los resultados que el sistema " Tutor " proporcione garantizarán la confiabilidad del análisis realizado y la capacidad del software creado, trayendo como consecuencia un mayor uso como herramienta de preparación

de exámenes de admisión al ciclo de bachillerato. Los tres tipos de factibilidad estudiada junto con las justificaciones que amparan al Sistema hacen que este sea factible.

MEDIOS

Alternativas con las que cuenta actualmente el alumno:

Remitirse a los apuntes y bibliografía recopilados durante su educación media básica.

Resolver las guías de estudio proporcionadas por las instituciones de educación media superior.

Asistir a cursos de regularización y de preparación para exámenes de selección para aspirantes a nivel medio superior.

Contratación de profesores particulares.

PROPUESTA DEL MÉTODO DE SOLUCIÓN

Elaboración de un sistema tutorial basado en la presentación de reactivos organizados por temas y áreas, con solución por opción múltiple y reafirmación conceptual al momento de la evaluación del reactivo, que ofrezca al alumno la posibilidad de obtener orientación que le ayude a responder al cuestionamiento, aclarando al mismo tiempo sus dudas logrando fortalecer su conocimiento y familiarización con exámenes de este tipo. Implementación de una *Interface* con el usuario de fácil manejo, convenientemente a través de *menús*, que le permitan el rápido acceso a los servicios proporcionados por el sistema. Almacenamiento de reactivos, respuestas e información conceptual en bases de datos bien estructuradas, que le den dinamismo al sistema y aseguren la congruencia y confiabilidad de la información presentada. Implementación de mecanismos de evaluación del rendimiento del aspirante, tanto en la preparación cotidiana (comparado con sesiones previas o con los resultados de otros aspirantes), como en la resolución de un examen de tipo real, que le permita conocer su nivel con

respecto al exigido por las escuelas de enseñanza media superior. Presentación atractiva del sistema, pudiéndose basar en la combinación de colores y posiblemente en la presentación de imágenes alusivas a los temas.

PRIORIDADES

- Crear software de calidad.
- Que sea un método de estudio atractivo y motivante.
- Que el sistema proporcione flexibilidad para estudiar por materia específica y adaptarse a las necesidades de cada aspirante.
- Suministrar medios para la evaluación del avance en el estudio.

Establecimiento de compromisos entre objetivos antagónicos o conflictivos

El objetivo de crear una herramienta que resulte atractiva, debe cumplirse en la medida en que no vaya en detrimento de aquel en que se pretende hacerla accesible al mayor número posible de estudiantes. En la actualidad, el avance en el *hardware* y *software* nos permite contar con recursos que podrían hacer muy atractiva la presentación del sistema, tales como ambientes gráficos, efectos de sonido e incluso la utilización de ambientes multimedia; sin embargo el elaborar un sistema que explotara tales elementos lo limitaría en el sentido de que solo podría ser utilizado por aquellos aspirantes que contaran con tan aún sofisticados recursos.

II.3 ESPECIFICACIÓN DEL SISTEMA

ESTABLECIMIENTO DE LAS NORMAS DE CALIDAD Y RESULTADOS ESPERADOS

Calidad del Software

El problema de la gestión de la calidad no es que la gente no sepa sobre ella. El problema es lo que creen que saben...

Salvando las diferencias, la calidad tiene mucho en común con el sexo.

Todo el mundo lo quiere (bajo ciertas condiciones por supuesto). Todo el mundo cree que lo conoce (incluso aunque no quieran explicarlo). Todo el mundo piensa que su ejecución solo es cuestión de seguir las inclinaciones naturales (después de todo, nos las arreglamos de alguna forma). Y, por supuesto, la mayoría de la gente piensa que los problemas en estas áreas son producidos por otros (como si solo ellos se tomaran el tiempo para hacer las cosas bien).[PRE88]

Calidad del Software. Es la concordancia con los requerimientos funcionales y de rendimiento explícitamente establecidos, con los estándares de desarrollo explícitamente documentados y con las características implícitas que se espera de todo software desarrollado profesionalmente.

Puntos importantes:

- 1) Los requerimientos del software son los fundamentos desde los que se mide la calidad. La ausencia de concordancia con los requerimientos es falta de calidad.
- 2) Los estándares especificados definen un conjunto de criterios de desarrollo que guían la forma en que se aplica la ingeniería del software. Si no se siguen esos criterios, casi siempre se dará una falta de calidad.
- 3) Existe un conjunto de requerimientos implícitos que a menudo no se mencionan (p.ej. el contar con un buen mantenimiento). Si el software se

ajusta a sus requerimientos explícitos pero falla en alcanzar los requerimientos implícitos, la calidad del software queda en entredicho.

Factores que determinan la calidad del software

Los factores que afectan a la calidad del software se pueden clasificar en dos grandes grupos:

- 1) Factores que pueden ser medidos directamente (errores, unidades de tiempo).
- 2) Factores que sólo pueden ser medidos indirectamente (facilidad de uso, mantenimiento del sistema).

Los factores de la calidad del software se centran en tres aspectos importantes : sus características operacionales, su capacidad de soportar los cambios y su adaptabilidad a nuevos entornos.

Factores de calidad del software

Objetividad. Es el grado en que un programa satisface sus especificaciones y consigue los objetivos planteados por el cliente.

Fiabilidad. El grado en que se puede esperar que un programa lleve a cabo sus funciones con la precisión requerida.

No hay duda que la fiabilidad de un programa de computadora es un elemento de su calidad en general. Si un programa falla frecuentemente en su funcionamiento, no importa si el resto de los factores de calidad son aceptables. La fiabilidad del software se define en términos estadísticos como "la probabilidad de operación libre de fallos de un programa de computadora en un entorno determinado y durante un tiempo específico".

Eficiencia. La cantidad de recursos de computadora y de código requeridos por un programa para llevar a cabo sus funciones.

Seguridad. La disponibilidad de procesos que controlen o protejan los programas o datos, y que permitan restringir el acceso a determinadas partes del sistema.

Integridad. Concierno a la confiabilidad de la información, refiriendonos a que esta debe ser veraz, es decir debe haber congruencia entre los datos reales y la información almacenada en el sistema.

Facilidad de uso. El esfuerzo requerido para aprender a trabajar con el sistema.

Facilidad de mantenimiento. El esfuerzo requerido para localizar y arreglar un error en un programa.

Flexibilidad. El esfuerzo requerido para modificar un programa operativo.

Facilidad de prueba. El esfuerzo requerido para probar un programa de forma que se asegure que realiza una función requerida.

Portabilidad. El esfuerzo requerido para transferir el programa desde un hardware y/o un entorno de software a otro.

Reusabilidad. El grado en que un programa (o partes de un programa) se puede volver a usar en otras aplicaciones. Esto va relacionado con el empaquetamiento y el alcance de las funciones que realiza el programa.

Actividades de la garantía de calidad del software

La garantía de calidad del software comprende una gran variedad de tareas asociadas con siete actividades principales:

- 1) Aplicación de metodologías técnicas.
- 2) Realización de revisiones técnicas formales.
- 3) Prueba del software.
- 4) Ajuste de los estándares.
- 5) Control de cambios.

6) Mediciones.

7) Registro y realización de informes.

La calidad del software debe estar diseñada en el sistema; no es algo impuesto a posteriori. Por esta razón la garantía de calidad del software comienza realmente con un conjunto de herramientas y métodos técnicos que ayudan al analista a conseguir una especificación de alta calidad y un diseño de alta calidad.

Una vez que se ha creado una especificación (o prototipo) y un diseño, debe ser garantizada su calidad. La actividad que permite garantizar la calidad es la revisión técnica formal (RTF), esta es una reunión del personal técnico con el único propósito de descubrir problemas de calidad. En muchas situaciones, se ha visto que las revisiones son tan efectivas como lo es la prueba para descubrir defectos en el software.

La prueba del software combina una estrategia de múltiples pasos con una serie de métodos de diseño de casos de prueba que ayudan a asegurar una efectiva detección de errores. Muchos grupos de desarrollo de software usan la prueba del software como una "red de seguridad" para la garantía de calidad. Esto es, asumen que mediante la prueba descubrirán la mayoría de los errores, mitigando así la necesidad de otras actividades de garantía de calidad del software.

El grado de aplicación de ajuste de los estándares varía de acuerdo a cada uno de los grupos de desarrollo. En muchos casos, los estándares vienen dados por los clientes o por mandamientos de regulación. En otras situaciones, los estándares vienen automáticamente impuestos. Si existen estándares formales, se debe establecer una actividad de garantía de calidad de software para garantizar que sean seguidos. La garantía de seguimiento de estándares puede ser llevada a cabo por los encargados del desarrollo del software como parte de una revisión técnica formal, y en situaciones en que se requiera una verificación del seguimiento independiente se llevará a cabo por los analistas encargados de realizar dichas tareas.

Una de las principales amenazas para la calidad del software es causada por los cambios. Cada cambio realizado sobre el software en potencia, puede introducir errores o crear efectos laterales que propaguen errores. El proceso de control de cambios contribuye directamente a la calidad del software al formalizar las peticiones de cambio, evaluar la naturaleza del cambio y controlar el impacto del cambio. El control de cambios se aplica durante el desarrollo del software y posteriormente, durante la fase de mantenimiento del software.

La actividad de medición es una actividad integral para cualquier disciplina de ingeniería. Un fin importante de la garantía de calidad del software es perseguir la calidad del software y garantizar la posibilidad de cambio metodológico procedimental para mejorar el software. Para conseguir esto, se deben recolectar métricas del software.

El registro de información y la generación de informes obtenidos de las revisiones, auditorías, control de cambios, pruebas, y otras actividades de la garantía de calidad de software dan la pauta para la mejor toma de decisiones que conllevan a la calidad del software.

Concepto de calidad

Su fin es producir artículos que satisfagan los requisitos de los consumidores, tomando en cuenta que las sugerencias de éstos varían frecuentemente. Los requisitos del consumidor son de primordial importancia. Los fabricantes de software deben estudiar las opiniones y requisitos de los consumidores y tomarlos en cuenta al diseñar, manufacturar y vender sus productos además de considerar el nivel de adquisición de los habitantes donde se lanzará el producto.

Puntos Básicos para el Control de Calidad

Mejorar la calidad mediante el Ciclo: P.H.V.A. ; Planear, Hacer, Verificar y Actuar.

Importancia de captar la dispersión en las estadísticas.

Control de procesos mediante el empleo de cuadros de control y cómo aplicarlos.

Hacer Control de Calidad significa

- 1.- Emplear el Control de Calidad como base.
- 2.- Hacer el Control Integral de costos, precios y utilidades.
- 3.- Controlar la cantidad (volumen de producción, de ventas y de existencia), así como las fechas de entrega.

Factores Importantes

- Control de Procesos.
- Encontrar causas de los efectos y fallas.
- Tener siempre presente los requerimientos del mercado y demanda.
- Hacer uso de la Estadística Aplicada (Muestreo). [ISH86]

La crisis del Software

La crisis del Software se refiere a un conjunto de problemas encontrados en el desarrollo del software de computadoras. Los problemas no están limitados al software que "no funciona adecuadamente". Sino que la crisis del software abarca los problemas asociados con: cómo desarrollar el software, cómo mantener un volumen creciente de software existente y cómo podemos esperar satisfacer la demanda creciente de software.

Problemas

La crisis del software se caracteriza por muchos problemas, pero los responsables del desarrollo del software se concentran sobre los aspectos de "fondo":

- 1) La planificación y estimación de costo es frecuentemente muy imprecisa.
- 2) La "productividad" de la gente del software no corresponde con la demanda de sus servicios.

3) La calidad del software no llega a ser a veces ni adecuada.

Al ser considerada nuestra aplicación como una innovación, no contamos con parámetros directos de comparación, así que compararemos la aplicación con otras aplicaciones de tipo windows que cumplen con las normas impuestas por IBM , para el desarrollo de aplicaciones windows, es decir , para el desarrollo del sistema se respetarán las normas brindadas inherentemente por el lenguaje Visual BASIC.

Calidad de la interfaz

La calidad no se basa en una lluvia de ideas, porque podríamos incurrir en el error de omitir alguna norma que proporcione calidad a nuestro software. Por lo cual necesitamos establecer un marco teórico y seleccionar una metodología que garantice la obtención de software de calidad. Para la interfaz de nuestro programa se utilizarán las siguientes normas :

Consistencia. Dentro de la aplicación la presentación de las diferentes pantallas será similar y con una constitución homogénea que no resulte extraña al usuario durante la ejecución del programa. Y respetará una serie de convenciones que han sido creadas para facilitar la labor del usuario.

Focalización. Esta norma establece que en todo momento el usuario debe de saber que parte del programa esta utilizando.

Estímulos aversivos. Esta norma establece que se deben de evitar (en todo momento de la ejecución) situaciones que presionen de una manera innecesaria al usuario.

El tener que seleccionar una opción entre veinte y que el programa brinde solamente cinco segundos para decidir, este es un muy buen ejemplo de como no actuar.

Retroalimentación. Esta norma exige que en todo momento se mantenga al usuario informado de lo que esta sucediendo en el sistema, es decir, si esta

ejecutando algún proceso, si sucedió un error, si se solicita una entrada de información , etc.

P.ej. Existen diversos sistemas en los cuales al realizar alguna operación de lectura o escritura a disco se visualiza una "línea" girando, representando movimiento y por lo tanto que esta realizando "algo".

Presentación de datos. Esta norma se utiliza para establecer un estándar para las propiedades de las entradas de información.

p.ej. Color de fondo de las pantallas, tipo de *Font*, tamaño del *Font*, estilo del *Font*, etc.

RESULTADOS ESPERADOS

Obtener software de calidad

Que los exámenes aplicados trasciendan de una manera significativa en la preparación del usuario en los temas de estudio.

Que el usuario obtenga un nivel adecuado de conocimientos en las diferentes áreas de estudio respecto al nivel exigido, por medio de la utilización frecuente del sistema.

De acuerdo a la opción de auto evaluación (Revisar calificaciones) le permita interpretar de una manera sencilla los índices del aprendizaje.

DEFINICIÓN DE LA PRECISIÓN CON QUE DEBEN SER ALCANZADOS LOS OBJETIVOS

El análisis de sistemas es el conocimiento de situaciones y no la solución de los problemas. Los buenos analistas , por lo tanto, hacen hincapié en la investigación para identificar cuales son los objetivos que deben de alcanzarse para la creación de un sistema.

Los estudios de sistema son el resultado de una evaluación para conocer cómo funcionan los métodos actuales. Estudiar el funcionamiento del sistema actual es indispensable para la recopilación de información e identificación de los objetivos que se pretende alcanzar; los analistas estructuran una investigación y buscan respuestas a las siguientes cuatro preguntas:

- ¿Cuál es el proceso básico ?
- ¿Qué datos se producen durante este proceso ?
- ¿Cuáles son los límites impuestos por tiempo y cantidad de trabajo ?
- ¿Qué controles de rendimiento se utilizan ?

Entender el proceso

Se empezará con lo básico. Se harán aquellas preguntas que proporcionarán, cuando se contesten, un antecedente de los datos fundamentales y de las descripciones del sistema. Las siguientes preguntas ayudan a adquirir el conocimiento necesario:

- ¿Cuál es el propósito de esta actividad ?
- ¿Cuáles son los pasos que se realizan ?
- ¿Quién los ejecuta ?
- ¿Dónde se realizan ?
- ¿Cuánto tiempo consumen ?
- ¿Con qué frecuencia se realizan ?
- ¿Quién utiliza la información resultante ?

Para el caso del sistema " Tutor " ¿Dónde se comenzaría ? a continuación se listan respuestas breves.

- ¿Cuál es el objetivo central del sistema " Tutor " ?

Proporcionar un método alternativo de estudio a los alumnos para la preparación de exámenes de admisión. El sistema " Tutor " debe ser un método atractivo que permita al alumno autoevaluarse y adquirir el nivel académico requerido.

¿ Cuáles son los pasos que se realizan ?

Consisten en el conocimiento de las áreas de estudio que son evaluadas y sus correspondientes temarios. Posteriormente se determina la manera en que se va estudiar, ya sea a través de guías de estudio, cursos de regularización o auto estudio a través de bibliografías, libros o apuntes.

¿ Dónde se realizan ?

La preparación hacia un examen de admisión se lleva a cabo en bibliotecas, en el hogar, en escuelas, etc. realmente esta respuesta es abierta para el caso del sistema " Tutor ".

¿ Quién los ejecuta ?

Propiamente como el sistema lo determina, los aspirantes son quienes llevan a cabo estas tareas.

¿ Cuánto tiempo consumen ?

El tiempo que se consume depende directamente de la confianza que el aspirante adquiera para la preparación del examen de admisión.

¿ Con qué frecuencia se realizan ?

Cada que se realice un examen de admisión.

¿ Quién utiliza la información resultante ?

El alumno ya que este podrá evaluarse a sí mismo.

En el sistema " Tutor " resulta sencillo poder responder estas preguntas; sin embargo, como toda disciplina en el análisis de sistemas deben ser siempre bien identificadas para poder lograr el alcance de los objetivos generales para cualquier sistema que pretenda desarrollarse. En cualquier caso estas preguntas proporcionan un entendimiento más amplio del proceso.

Ahora bien, no basta con entender solamente el proceso, se tiene que tener en cuenta cada uno de los puntos que a continuación se dan y que proporcionan precisión en el alcance de objetivos.

Oganización y control de actividades en un proyecto de análisis

- Organización básica
- Planeación básica
- Planeación de actividades enfocadas a la calidad
- Esquema de planeación y diseño conceptual

Elementos primarios de un proyecto de software

- **Personal:** Dentro de un proyecto de software existe siempre la necesidad de disciplina y comunicación entre líderes de proyecto, analistas y programadores.
- **Procesos:** Es la manera como un proyecto de desarrollo de sistemas o cualesquiera de sus partes integrales va a ser planeada, desarrollada o monitoreada
- **Actividades:** Es una porción de trabajo dentro de un proyecto que típicamente se asigna a alguien, que requiere de una entrada, proceso, salida, duración y una programación en un esquema de planeación.

Premisas de la disciplina dentro de la organización

- Es importante que exista un ambiente en el que todo mundo sepa que hacer.
- La disciplina es el elemento que mantiene al proyecto de una manera consistente.
- Los procesos y metodologías implementadas solo pueden ser mantenidas con una estricta disciplina.

En pocas palabras un proyecto no existirá y no se lograrán alcanzar los objetivos planteados si no existe disciplina.

"La disciplina es el alma de un equipo. Hace grande a equipos pequeños, procura éxito al débil y da aprecio a todos".

Reconocimiento a una organización disciplinada

- Disciplina es el acto de encaminar un patrón de comportamiento deseado.
- La disciplina correctamente ejercitada por un líder es: qué se está haciendo y cómo se está haciendo.

Para implementar una disciplina que conduzca al logro de los objetivos del proyecto deben proponerse metas realistas, obtención de compromisos, monitoreos del avance de acuerdo al plan de trabajo establecido y por último forzar los compromisos.

Aspectos para incrementar una buena comunicación

- Respeto al individuo.
- Admitir equivocaciones.
- Realizar Compromisos.
- Ejercitar tolerancia.
- Preguntar a otros si existen dudas.
- Mantener informados a todos, sin dar sorpresas.
- Mantener una educación en la comunicación.
- Saber escuchar.
- Preguntar, no asumir.
- Conocer que se espera de cada uno de los participantes.

DIAGRAMA DEL SISTEMA

- Flujo de proceso

El flujo de los procesos asignados por eventos al sistema se basará principalmente en la "carta de navegación" (como la nombramos), que describe las relaciones entre los diferentes controles y formas del sistema, haciendo uso de las propiedades inherentes a cada uno de ellos y que permitirán dar una continuidad lógica al manejo de la información y de los procesos. El flujo de procesos específico para la realización de los diferentes módulos se basará en el siguiente menú que representa los niveles jerárquicos de los procesos que consideramos necesarios para la implementación del sistema, es decir, el flujo de procesos tendrá una relación uno a uno con los eventos asociados a los controles correspondientes a lo largo del programa y dependiendo también de la selección deseada durante la ejecución del programa.

- &Archivo
 - &comenzar &Nuevo examen
 - &Continuar examen suspendido
 - &Salir
- &Calificar
 - &Cargar calificaciones
 - &Imprimir calificaciones
 - &Revisar calificaciones
 - &Salir del menú
- &Información
 - &Acerca de
- &Herramientas
 - &Panel de control
 - &Ejecutar proceso Win\DOS
 - &Calculadora aritmética
 - &Reloj
 - &Analógico
 - &Digital
 - &Ayuda

De lo anterior, la simbología es la siguiente:

Cada cuatro símbolos "-" significa un nivel jerárquico dentro del menú, el símbolo & acompañado por una letra mayúscula será la letra por la cual podremos acceder a dicha opción del menú en combinación de la tecla "Alt" como en la mayoría de los paquetes orientados a *menús* .

Del *menú* anterior podemos observar lo siguiente: en el primer nivel jerárquico hay cuatro opciones que son:

-Archivo: Esta opción del menú nos permitirá usar los procesos de comenzar un nuevo examen, continuar uno suspendido o salir del sistema, por lo tanto, esta opción a su vez contiene tres opciones que son:

- comenzar nuevo examen: Al hacer esta selección, tendremos que decidir entre ejecutar un proceso que realice un examen de tipo global o por sección, en caso de ser por sección se debe ejecutar un proceso que permita decidir la materia que se desea examinar, a continuación (también para el caso global) se requiere de un proceso que genere las preguntas correspondientes al usuario.
- continuar examen suspendido: Esta opción llamará a un proceso que permita seleccionar de una lista de exámenes suspendidos personalizados.
- salir : Esta opción llamará a un proceso que nos permitirá abandonar la aplicación.

-Calificar : Esta opción nos permitirá cargar de una lista un examen específico, para así poder calificarlo o imprimir los resultados del mismo, de aquí que esta opción cuente con los siguientes *submenús*.

- **cargar calificaciones:** Ejecutaremos un proceso que cargue dos archivos, uno que contenga la información concerniente al tipo de examen efectuado, tiempo utilizado, nombre del usuario, etc... y otro archivo que contiene las respuestas del examen, los identificadores de las preguntas con que se examinó al usuario y las respuestas de éste.
- **revisa calificaciones:** Es el proceso que se encarga de revisar los exámenes realizados.
- **imprimir calificaciones:** Es el proceso que genera los reportes de las calificaciones.

-Información : En las aplicaciones Windows es importante poseer un proceso que nos proporcione información acerca del autor y la versión de la aplicación.

-Herramientas : Al seleccionar esta opción podremos acceder el submenú que nos permitirá utilizar cinco herramientas que harán el entorno del usuario más amigable:

- **panel de control:** Es un proceso que brinda la facilidad para modificar el tamaño de la ventana donde ejecutaremos la aplicación y el color de fondo con el que deseemos trabajar (unicamente dentro del menú principal).
- **ejecutar procesos :** Es un proceso que nos permite ejecutar archivos BAT, EXE o COM desde la aplicación.
- **calculadora aritmética :** Es un proceso que nos permite realizar operaciones básicas de una manera sencilla.

- reloj : Es un proceso que nos permite *visualizar* la hora del sistema .
- ayuda :Esta opción contará con un submenú que permitirá acceder los *menús* de ayuda, es decir, de cada uno de los temas de las materias a examinar, haciendo uso de los archivos de hipertexto.

Al haber definido las partes anteriores estamos particionando los problemas que debemos solucionar, además de hacer una descripción a grosso modo de los procesos necesarios para el desarrollo del sistema .

En la "carta de navegación" se muestran las relaciones y vínculos existentes entre cada uno de los procesos mencionados anteriormente.

- Flujo de información

Hasta este punto hemos identificado los problemas , sus causas y hemos propuesto reglas y procedimientos para resolverlos. Cada momento del análisis tiene una única notación y punto de vista. Sin embargo, todos los métodos de análisis están relacionados por un conjunto de principios fundamentales:

1. El dominio de la información, así como el dominio funcional de un problema que debe ser representado y comprendido.
2. El problema debe subdividirse de forma que se descubran los detalles de una manera progresiva o jerárquica.
3. Deben desarrollarse las representaciones lógicas y físicas del sistema.

Aplicando estos principios , enfocamos el problema sistemáticamente. Se examinó el dominio de la información de forma que pueda comprenderse su función completamente. La partición se aplica para reducir la complejidad. La visión física y lógica del *software*, es necesaria para acomodar las

relaciones lógicas impuestas por los requerimientos de procesamiento, y las relaciones físicas impuestas por otros elementos del sistema.

El dominio de la información

Todas las aplicaciones del *software* pueden llamarse de una manera colectiva: procesamiento de datos. Este término contiene la clave de lo que entendemos por requerimientos del software. El software se construye para procesar datos; para transferir datos de una forma a otra; esto es, para aceptar entrada, manipularla de alguna forma y producir una salida. Este establecimiento fundamental de los objetivos se aplica tanto para sistemas en tiempo real, software en *batch* (lotes), y aun en los sistemas desarrollados bajo la filosofía por eventos, como lo es el sistema " Tutor ". El dominio de la información contiene tres visiones diferentes de los datos que son procesados por los programas de computadora: 1)el flujo de información ; 2)el contenido de la información, y 3)la estructura de la información. Para comprender el dominio de la información deben considerarse cada una de las tres partes.

El flujo de la información representa la manera en que los datos cambian conforme pasan a través de un sistema. En la siguiente figura II.3.1, la entrada se transforma en datos intermedios y más adelante se transforma en la salida. A lo largo de este camino de transformación (o caminos), pueden introducirse datos adicionales de los archivos de bases de datos existentes (p.ej., un archivo de disco o un *buffer* de memoria). Las transformaciones que se aplican a los datos son funciones o subfunciones que un programa debe ejecutar. Los datos que se mueven entre dos transformaciones (funciones) definen la interface de cada función. El contenido de la información representa los elementos de datos individuales que componen otros elementos mayores de información. Por ejemplo, un registro de la base de datos de las preguntas de examen o un registro donde se almacena la información de un examen suspendido o las calificaciones de este, son elementos de información que están formados por cadenas de caracteres y números . El contenido de un registro queda definido por los elementos que lo constituyen.

Para comprender el procesamiento que hay que aplicar a un registro de los mencionados debemos comprender primero la información contenida y no tratarla a está como solo un conjunto de números y letras.

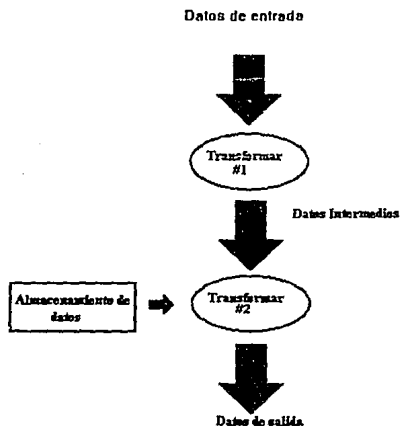


FIGURA II.3.1

La estructura de la información representa la organización lógica de los distintos elementos de datos.

Tomando en cuenta lo anterior, a continuación se muestran los enunciados de los requerimientos y comportamiento funcional que determinaran finalmente las relaciones existentes entre las entradas, los procesos y las salidas.

Los requerimientos del programa, manifestados mediante el dominio de la información, requerimientos funcionales y de comportamiento alimenta la fase de diseño. El diseño de datos se enfoca sobre la definición de la estructura de datos. El diseño arquitectónico define las relaciones entre los

principales elementos estructurales del programa. El diseño procedimental transforma los elementos estructurales en una descripción procedimental del software. Para integrar y validar el software se genera y prueba el código fuente. Aunque entre las fases de desarrollo y pruebas absorben más del 75% del costo de la ingeniería de *software*(excluyendo el mantenimiento), es aquí donde tomamos las decisiones que afectarán finalmente el éxito de la implementación del programa e igualmente importante a la facilidad con que el programa será mantenido .

La importancia del diseño del software puede establecerse con una única palabra: calidad. El diseño es la parte donde se asienta la calidad del desarrollo del sistema. El diseño nos da las representaciones del software que pueden establecerse para conseguir un producto con calidad. El diseño es la única forma mediante la que podemos traducir con precisión los requerimientos del cliente en un producto o sistema acabado. El diseño de programas sirve como base para todos los pasos de desarrollo y fase de mantenimiento . Sin el diseño, nos arriesgamos a construir un sistema inestable, un sistema que fallará cuando se realicen pequeños cambios, un sistema que puede ser difícil de probar, un sistema cuya calidad no puede ser establecida hasta más adelante en el proceso de ingeniería de software.

SELECCIÓN DEL EQUIPO DE CÓMPUTO PARA EL DESARROLLO DEL SISTEMA

Debido a los usuarios a los que estará destinado el diseño del " Tutor ", se ha pretendido trabajar dentro del entorno en el cual el sistema funcionará y será utilizado. Por tanto, el desarrollo del sistema ha decidido llevarse a cabo sobre computadoras personales con sistema operativo MS-DOS.

MS-DOS es actualmente el sistema operativo más usado para computadoras personales e incluye características que le dan la capacidad a una computadora personal para ser usada de una manera sumamente eficiente y sobre todo de una manera sencilla en comparación a otros equipos de cómputo.

Gran parte del diseño y desarrollo requiere de Windows, debiéndose considerar entonces que para la instalación de este software y versión 6 de MS-DOS se requiere un mínimo de 15.8 MB y un máximo de 18.3 MB de espacio en disco duro. La diferencia entre una instalación y otra radica básicamente en que para la mínima no se incluyen componentes adicionales (impresora y red) y para la máxima si.

Para la programación del sistema usamos el siguiente hardware :

- 200 MB en disco duro
- 8 MB de memoria RAM
- Unidad CD-ROM
- FD 1.2 MB 5.1/4"
- FD 1.4 MB 3.5 "
- Procesador pentium a 60 MHz

El siguiente equipo fue utilizado para la captura de las bases de datos de las áreas de estudio consideradas por el " Tutor ", y para la edición de textos relativos al sistema.

- 40 MB en disco duro
 - 1 MB de memoria RAM
 - FD 1.4 MB 3.5 "
 - Procesador 80-286 a 12 MHz
-
- 200 MB en disco duro
 - 8 MB de memoria RAM
 - FD 1.2 MB 5.1/4
 - Procesador 80386 a 25 MHz

El siguiente equipo se usó para cargar las bases de datos de las diferentes áreas de estudio, además de la edición de los archivos que se utilizarán en los menús de ayuda y para la edición de preguntas que requieren objetos de tipo "fórmula" y elementos gráficos en power point.

- 100 MB en disco duro
- 2 MB en RAM
- FD 1.2 MB 5.1/4"
- Procesador 80-386 a 25 MHz

El siguiente equipo se uso para cargar las bases de datos de las diferentes áreas de estudio, además de la edición de los archivos que se utilizarán en los menús de ayuda .

- 100 MB en disco duro
- 2 MB en RAM
- FD 1.2 MB 5.1/4"
- Procesador 80-386 a 25 MHz

- 130 MB en disco duro
- 6 MB en RAM
- FD 1.2 MB 3.1/2
- Procesador 80-386 a 25 MHz

Un requerimiento de hardware por parte de windows 3.1 para que trabaje en modo extendido es que el procesador de la computadora sea 80-386 o superior.

Es posible ejecutar windows en dos modos que son modo estándar y modo extendido. Cuando se teclaea win desde la línea de comando *Windows* se ejecuta de la manera más adecuada para el sistema (dentro de *Windows* se puede verificar en que modo se esta ejecutando en el menú de ayuda en la opción de administrador del sistema).Windows toma los siguientes criterios para determinar que modo utilizar.

- Windows arranca en modo estándar si se dispone de un procesador 80286 o mayor con al menos 640 Kb de memoria convencional y 384 Kb de memoria extendida.Windows también arranca en modo estándar en computadoras 80386 que tengan menos de 2 MB de memoria libre.

- Windows arranca en modo extendido si se tiene un procesador 80386SX o mayor con al menos 640 Kb de memoria convencional y más de 1024 Kb de memoria extendida . De cualquier modo, es posible que se requiera más memoria para usar características más avanzadas de Windows.

El equipo periférico que se utilizó es el siguiente:

- Impresora láser HP laserjet III
- Impresora Sharp JX-9500H emuladora HP laserjet IID
- Impresora Star XR-1520 emuladora Epson
- Scanner HP color +

La selección del equipo de cómputo para el desarrollo tomó como principal criterio la disponibilidad del equipo existente en las áreas de trabajo donde actualmente laboramos y la distribución de cargas de trabajo y tareas específicas que se asignó de acuerdo al poder de procesamiento de cada uno de los equipos y a su capacidad de memoria.

ENTORNO DE PROGRAMACIÓN

Se eligió para este proyecto el sistema operativo "DOS" en cualquiera de sus variantes superiores o iguales a 5.0 y el ambiente operativo MS-Windows 3.1, debido a que al ser un ambiente gráfico permite desarrollar con la paquetería y lenguajes elaborados para el mismo, interfaces gráficas en las que las funciones pueden asociarse a símbolos, lo que facilitará al usuario el aprendizaje del manejo del sistema " Tutor ". Además cuenta con herramientas para la implementación de ayuda relacionada al contexto, lo cual es particularmente útil en sistemas como el que estamos desarrollando.

La filosofía Windows

Windows no es tan sólo un software que se ejecuta sobre el sistema operativo, sino que en cierta medida lo reemplaza, dando al usuario un entorno gráfico amigable para su interacción con la parte de *hardware*. A nivel interno posee toda una serie de características destinadas tanto a facilitar su uso, como a mejorar las presentaciones del sistema operativo.

Las principales ventajas que nos ofrece este entorno son:

- Homogeneidad del entorno. (CUA, Common User Acces). La inmensa mayoría de las aplicaciones Windows presentan al usuario una interface estandarizada, de forma que su aspecto es siempre similar y su utilización idéntica, lo que facilita enormemente el aprendizaje.
- Intercambio de datos entre aplicaciones. En el entorno de Windows es común y corriente realizar fácilmente el intercambio de datos entre aplicaciones mediante un método estandarizado, como por ejemplo el uso del portapapeles o el DDE (Dynamic Data Exchange).
- Gestión de toda la memoria. Con la aparición del procesador 80286, se introdujeron nuevos modos de gestión de memoria, que ha aprovechado Windows ampliamente. De esta forma cualquier aplicación Windows puede acceder a toda la memoria que se disponga.
- Independencia del *hardware*. Windows permite ejecutar una misma aplicación en múltiples configuraciones de *hardware*, por ejemplo, con distintas tarjetas gráficas, distintas impresoras, etc. Esto es debido a que las aplicaciones no necesitan administrar los distintos recursos disponibles, sino que se limitan a solicitar a Windows la gestión necesaria en cada momento y es éste el que con ayuda de los controladores adecuados ("Drivers" proporcionados por el fabricante del *hardware*) se encarga de realizarlo.
- Multitarea. Dentro del entorno Windows se pueden mantener abiertas a la vez varias ventanas y en cada una de ellas puede encontrarse una aplicación distinta.

Cabe añadir que la multitarea Windows no es, al menos de momento, una multitarea real, en la que el sistema operativo determina el tiempo de CPU que se asignará a cada aplicación, sino que es necesaria la colaboración amistosa de todas las aplicaciones, que deben de liberar el uso del CPU a intervalos apropiados para que sea otra aplicación la que se utilice.

Todas estas características también proporcionan una serie de ventajas para el desarrollo de aplicaciones Windows, como son el no tener que realizar una programación específica de los periféricos de entrada y salida actuales o que se utilicen en un futuro, acceso a más memoria, un rápido aprendizaje de la utilización de su aplicación por el usuario y con resultados más elaborados, etc.

La programación en Windows

La aparición de *Windows* no sólo conmovió la forma de interactuar del usuario con la máquina sino que introdujo una nueva filosofía también para los programadores.

Una de las principales diferencias que presenta la programación en Windows viene impuesta por la multitarea propia del sistema, ya que aquí no es una sola aplicación que se está ejecutando sino que son varias; como mínimo el propio Windows, además debe preverse la contingencia de que se trabaje simultáneamente con más aplicaciones. Es por eso que no puede hacerse un uso exclusivo de los recursos del sistema. Esto es evidente si pensamos que la pantalla no es exclusiva del programa sino que la salida se limita a una ventana, aún cuando ésta ocupe toda la pantalla. Al existir varias aplicaciones en ejecución y la necesidad de que los recursos del sistema se compartan entre ellas, no puede ser el programa el que administre los recursos, sino tan solo emitir peticiones de adjudicación de los mismos, por ejemplo, supongamos un *click* o una pulsación del teclado, la aplicación no puede permanecer esperando la pulsación o el *click*, observando continuamente el estado del teclado, como en *MS-DOS*. En el caso de aplicaciones Windows, aún cuando existan varias, únicamente una es la activa y ella irá dirigida a la pulsación o al *click*. El sistema necesita pues *algo* que se encargue de recoger la pulsación o el *click* y dirigirlo a la aplicación adecuada, esto es, Windows.

El propio entorno Windows es el que se encarga de vigilar el sistema y, frente a cualquier eventualidad, avisar a la aplicación adecuada. Estos avisos son lo que se denomina sistema de mensajes de Windows, siendo uno de los puntos primordiales de toda la filosofía Windows. En resumen, ante

cualquier evento que se produzca en el sistema, Windows genera un mensaje dirigido a la aplicación correspondiente.

Con todo esto se ha introducido un nuevo concepto: la interactividad de la aplicación con el entorno.

En una programación tradicional el código es lineal; unas sentencias se ejecutan tras otras sin más posibilidad que esperar entrada de datos en ciertos puntos, continuando su ejecución con posterioridad. En la programación bajo Windows la aplicación consiste en una serie de subrutinas que entran en acción cuando se produce un evento determinado, que genera el correspondiente mensaje.

A lo largo de estos párrafos se ha empleado bastante el termino *evento*, ¿Pero qué es en realidad un evento dentro de la programación Windows?. Cualquier suceso provocado por el usuario o el propio sistema y que es capaz de generar el correspondiente mensaje, por ejemplo un doble *click*, una pulsación de teclado, un *click*, un *gofocus*, un *losthfocus*, un movimiento del *mouse*, cambio de tamaño de una ventana, etc. Todas estas peculiaridades originan, como es lógico, toda una serie de ventajas e inconvenientes que a continuación se resumen.

La primera de las ventajas que nos aporta es tener acceso de una forma sencilla a la generación de interfaces de usuario de gran calidad y estandarizados. Como ya se mencionó, Windows provee de independencia del *hardware*, lo cual también es válido para la programación, así no es necesario programar directamente para tipos de *hardware* particulares.

Otra ventaja para las aplicaciones desarrolladas es la posibilidad que brinda el entorno de gestionar completamente la memoria, de forma que se tiene acceso a toda ella.

Como contrapartida, este nuevo tipo de programación también plantea complicaciones adicionales como son el preparar la aplicación para que en todo momento se encuentre dispuesta a satisfacer las peticiones que le realice el usuario.

Otro inconveniente, lo impone la multitarea , puesto que hay que tener en cuenta que se debe colaborar con el resto de las aplicaciones y no acaparar los recursos de forma innecesaria . Esto implica un control cuidadoso del desarrollo.

El tiempo que retengamos el control cada vez que se procesa un mensaje también es importante puesto que hasta que no finalizamos estamos impidiendo la ejecución del resto de las aplicaciones, ya que Windows es un sistema multitarea "*nonpreemptive*" (no es el procesador el que distribuye los tiempos de ejecución).[VIS93]

Como lenguaje de programación seleccionamos Visual BASIC versión 3.0 edición *professional*, que es un lenguaje de programación dirigida por eventos, lo cual también resulta muy adecuado al sistema que estamos elaborando, las ventajas proporcionadas por este lenguaje son:

Cuenta con múltiples facilidades para la elaboración de interfaces gráficas atractivas y eficientes.

Proporciona un lenguaje formal de programación.

Cuenta con herramientas para el manejo directo de bases de datos.

Facilita la incrustación de objetos provenientes de otras aplicaciones, tales como gráficos, imágenes, textos, tablas,etc...

Para el almacenamiento de la información elegimos el formato conocido como DBF (data base file) el cual es un estándar para la plataforma XBase en la materia de bases de datos, que no es más que la parte en común en que se basan los paquetes comerciales *DBase*, *Summer 87*, *Clipper* y *Visual Objects* y funciona en cualquiera de dichos paquetes. Para la obtención de dicho formato utilizamos *Dbase IV* ya que es un manejador de bases de datos fácil de usar que proporciona herramientas para la captura, consulta y modificación de la información que no requiere de programación especial para el efecto y considerado como un manejador de bases de datos de mediano alcance con el que todos los integrantes del equipo estamos

familiarizados. Decidimos usar dicho manejador de datos porque la cantidad de información a almacenar es considerable. Por las características del sistema, no se requiere de un esquema de seguridad sofisticado, ya que la información que manejamos es de dominio público, por otra lado esta es una de las *bases de datos* que es fácilmente manejable a través de Visual BASIC.

El editor Word para Windows 2.0 se eligió tanto para la elaboración de la tesis como para la documentación propia del sistema y los *menús* de ayuda integrados al sistema que cuentan con ayuda relacionada al contexto o hipertexto, ya que presenta las ventajas de ser compatible con el *software* elegido anteriormente, y de contar con herramientas como corrector ortográfico y la librería de sinónimos, así como diccionarios personalizados. Tiene la capacidad de acceder a los *fonts* que maneja Windows, y facilita la incrustación de objetos tales como gráficos, ecuaciones, imágenes e incluso pantallas, lo que nos ayudará a elaborar una documentación clara y con buena presentación.

Para la elaboración de gráficos se seleccionó el paquete *Power Point 2.0*, ya que su manejo es muy sencillo, y sus gráficos pueden ser incrustados fácilmente en otras aplicaciones para Windows, además maneja una gran variedad de formatos de archivos de imágenes, incluso aquellos generados por *scanners*. Este paquete es muy útil en la elaboración de presentaciones por computadora, lo que puede servirnos para la presentación de la tesis.

Otra razón importante en nuestra selección de *software*, es que tenemos acceso a los paquetes originales, la licencia para utilizarlos, y a toda la documentación distribuida con los mismos.

ENTORNO DEL USUARIO

Asumimos que el usuario del sistema " Tutor ", ha tenido (poco o nulo) contacto con computadoras, sin embargo, consideramos que en su vida cotidiana al haber utilizado aparatos electrónicos domesticos programables (videocaseteras, televisores, equipos modulares e incluso algunos juegos de vide y hornos de micro-ondas) que le ayudarán a familiarizarse con el sistema " Tutor ".

Por lo anterior lo más adecuado es la implementación de una interface amigable, que permita al usuario el manejo del sistema con un mínimo de conocimientos previos en computación.

Lo más recomendable parece ser una interface gráfica, en la que la navegación se realice completamente a través de menús, y la selección de opciones mediante el *mouse*, y entre botones de control que proporcionen a la vez una ayuda visual.

El ambiente más recomendable para el manejo de este tipo de interfaces es *Windows*, y la paquetería desarrollada para el mismo, debido a la naturaleza gráfica del sistema, y a la forma en que asocia imágenes a tareas específicas, creando una forma de comunicación basada en símbolos representativos (iconos).

Consideramos que es necesario que el sistema " Tutor " tenga un proceso de instalación muy sencillo, en el que el usuario no tenga que preocuparse (al menos en un principio) por especificar la configuración o personalización del sistema.

La ejecución del sistema debe también ser un proceso simple, por lo que se asociará al sistema un icono que el usuario pueda identificar fácilmente.

III. DISEÑO

III.1 DOCUMENTACIÓN DE LA ARQUITECTURA

LOS CONJUNTOS DE INFORMACIÓN

Definición de los conjuntos de información

Los conjuntos de información se dividen en tres tipos que son

1. Conjuntos de información de entrada
2. Conjuntos de información de salida
3. Conjuntos de información de bases de datos

Los conjuntos de información de entrada son todos aquellos elementos de información necesarios para la correcta operación de la programación y que usualmente se capturan a través de los diferentes dispositivos de lectura de los sistemas de cómputo, tales como unidades de discos magnéticos, etc.

En el sistema "Tutor" la información se maneja en tres grandes bloques que son:

1. Las bases de datos relacionadas con los reactivos

Como conjunto de información para los reactivos se generaron 10 bases de datos (Tablas para Bases de Datos Relacionales) que son :

Quimica
Matemati
Literatu
Hist_uni
Hist_mex
Geo_mex
Geo_gral
Fisica
Espanol
Biologia

Con la siguiente estructura

Nombre del campo	Tipo	Ancho	Índice
TEMA	Numérico	7	No
PREGUNTA	Memo	10	No
A	Caracter	70	No
B	Caracter	70	No
C	Caracter	70	No
D	Caracter	70	No
E	Caracter	70	No
RESPUESTA	Caracter	1	No
N_PASAJE	Numerico	4	No

2. Los archivos relacionados con exámenes suspendidos y exámenes por calificar.

[archivo].EXM Se genera cuando termina un examen (cuando expira el tiempo o se finaliza). Contiene la información correspondiente a :

La contraseña del examen, autor, tipo de examen, materia (si es por sección), si es examen suspendido, fecha y hora de realización.

[archivo].CAL Se genera cuando termina un examen (cuando expira el tiempo o se finaliza). Contiene la información correspondiente a :

Los números de pregunta, respuestas del usuario, es decir se usa para poder calificar un examen terminado.

[archivo].SUS Se genera cuando suspendemos un examen (cuando expira el tiempo o se finaliza). Contiene la información correspondiente a :

La contraseña del examen, autor, tipo de examen, materia (si es por sección), si es examen suspendido, fecha y hora de realización.

[archivo].CAS Se genera cuando termina un examen (cuando expira el tiempo o se finaliza). Contiene la información correspondiente a :

Los números de pregunta, respuestas del usuario, es decir, se usa para poder calificar un examen suspendido.

Estos archivos son considerados de E/S (entrada/salida) por lo que en ocasiones habrá la necesidad de abrirlos para escribir información en ellos y en ocasiones lo haremos para leerla, estas operaciones se realizan directamente en el disco duro.

3. Los archivos relacionados con el hipertexto

[archivo].RTF Son los archivos de cada uno de los temas que contienen los "hot spots" (saltos a "pop-up windows" o "jumps" a otros hipertextos y/o llamadas a macros). También pueden ser archivos de tipo "Contents" que asemejan un menú.

RTF significa "Rich Text Format" y es un formato aceptado por la mayoría de los paquetes comerciales de hojas de cálculo y procesadores de palabras.

[archivo].H Son los archivos que "vinculan" los archivos .RTF (valga la redundancia) de un tema específico con algún elemento de Visual BASIC; asociando un identificador numérico (también es posible con una cadena de caracteres) con la propiedad "HelpContextID" de algún control del lenguaje de programación.

[archivo].HPJ Son los archivos de "proyecto" que se compilan para generar un archivo de ayuda .HLP .

[archivo].HLP Son los archivos ejecutables del sistema de ayuda para el programa "Tutor"; es decir, archivos de tipo "Help for Windows" o "WinHelp"

Conjuntos de información de entrada

Los conjuntos de información de entrada del sistema "Tutor" son :

-Las Bases de Datos

Contienen propiamente los reactivos de los exámenes generados por el sistema "Tutor" en sus diferentes modalidades.

-Los archivos de ayuda relacionada al contexto (Hipertexto)

Contienen los diferentes temas que contempla el programa de estudio de cada una de las materias.

El proceso de lectura de la información de entrada del sistema " Tutor " es transparente para el usuario, lo cual facilita su operación.

Cada módulo del sistema " Tutor " espera una respuesta a todos los eventos asociados. Los eventos considerados dentro del sistema " Tutor " son los siguientes :

Click	GotFocus	KeyUp	Resize
DoubleClick	KeyDown	MouseMove	

-Los conjuntos de Información de salida

El sistema "Tutor" actúa de una manera interactiva, debido a la programación orientada a eventos y el ambiente operativo Windows.

El hipertexto que se maneja dentro del programa "Tutor" requirió de un diseño independiente, se basó en los temas de la guía de estudio más otros temas que consideramos relevantes. En el hipertexto se maneja la información de salida de una manera dinámica y brinda al usuario una navegación sobre los temas relacionados.

Los reportes de calificaciones los consideramos como un tipo de salida enfocada al usuario y se direcciona tanto a pantalla como a impresora, proporcionando al usuario una herramienta adicional para llevar un registro más formal de sus deficiencias o avances en su preparación.

PROCESOS O PROGRAMAS A DESARROLLAR Y COMPORTAMIENTO DINÁMICO

Del análisis de requerimientos del sistema definimos cuatro grandes módulos:

1. Archivo

Este módulo del menú nos permitirá usar los procesos de comenzar un nuevo examen, continuar uno suspendido o salir del sistema; por lo tanto, esta opción a su vez contiene tres opciones y las formas correspondientes que los componen son:

comenzar nuevo examen, continuar examen suspendido y salir, se utilizarán las siguientes formas:

Tipo_examen : Decide entre tipo de examen global o por sección

Examen_x_sección: Seleccionar la materia a examinar.

Tutor_o_Examen: Seleccionar entre realizar examen o ejercicios tipo tutor.

preguntas_global : Realiza todos los procesos necesarios para un examen de tipo global.

preguntas_gauge: Realiza todos los procesos necesarios al realizar un examen por sección en tipo tutor.

preguntas_x_seccion : Realiza todos los procesos para efectuar un examen por sección en tipo examen.

sal_suspendido: Efectúa los procesos referentes para salvar un archivo suspendido o terminado

salvar_suspendidos: Genera los archivos de exámenes suspendidos.

salvar: Genera los archivos de calificaciones.

2. Calificar : Esta opción nos permitirá cargar un examen específico de una lista, para así poder calificarlo o imprimir los resultados del mismo, de aquí que esta opción cuente con tres *submenús* y las formas que lo integran son:

carga_calif: Carga las calificaciones de los exámenes.

revisa: Revisa el examen correspondiente.

3. Información : En las aplicaciones Windows es importante poseer un proceso que nos proporcione información acerca del autor y la versión de la aplicación. La forma correspondiente es:

acer: despliega una forma con la información "Acerca de".

4. Herramientas : Al seleccionar esta opción podremos acceder el *submenú* que nos permitirá utilizar herramientas que harán el entorno del usuario más amigable. Sus formas son:

PanelCo: Despliega un panel de control.

Ejecuta: Ejecuta procesos .bak, .exe o .com.

Calcular: Permite realizar operaciones aritméticas.

Relo_analogico y Relo_digital: Permiten visualizar la hora del sistema.

Ayud: *menú* general de ayuda relacionada al contexto.

Al haber definido las partes anteriores estamos particionando los problemas que debemos solucionar ademas de hacer una descripción a grosso modo de los procesos necesarios para el desarrollo del sistema .

LAS RESPONSABILIDADES

La descripción del análisis de sistemas da una buena idea de lo que hace el personal dedicado a esta área. Para efecto de comprender el entorno de cómo influye el manejo de las responsabilidades dentro del ciclo de vida de un sistema, se presenta a continuación una lista de los conjuntos más comunes de responsabilidades que son asignadas.

Solamente el análisis de sistemas: Esta responsabilidad que es totalmente del analista, radica en conducir el estudio del sistema para conocer los hechos importantes en relación con la actividad del área que se planea sistematizar. El hincapié se hace en determinar los requerimientos de información y del proceso. Sus responsabilidades no incluyen el diseño de sistemas. (Analista de información.)

Análisis y diseño de sistemas: Aquí se lleva a cabo el estudio completo del sistema, pero también se tiene la responsabilidad del diseño del sistema. (Líderes de proyecto.)

Análisis, diseño y programación de sistemas: Esta gente dirige la investigación de sistemas, las especificaciones del diseño del desarrollo y programa el software para poner en marcha el diseño. (Analistas-programadores); el sistema " Tutor " fue diseñado y desarrollado de esta manera.

Los tres puntos anteriores nos muestran cómo deben de ser distribuidas las responsabilidades dentro de un equipo de trabajo dedicado al análisis, diseño y desarrollo de sistemas.

**ESTA TESIS NO DEBE
SALIR DE LA BIBLIOTECA**

Para el caso del sistema " Tutor ", la designación de responsabilidades está distribuida de la siguiente manera:

Analistas-programadores: Cuauhtémoc Domínguez Olvera
Armando Barragán Maya
Luis Alonso Vásquez García
Estela Jimenez Canchola

III.2 DISEÑO DEL PROGRAMA DE COMPUTADORA

DIAGRAMA DE ESTRUCTURA

Preguntas_x_Seccion	CtrlName	Caption
Etiqueta 1	Label1	
Etiqueta 2	Label2	
Etiqueta 3	Label3	
Etiqueta 4	Label4	
Etiqueta 5	Label5	
Etiqueta 6	Label6	
Etiqueta 7	Label7	
Etiqueta 8	Label8	
Etiqueta 9	Label9	
Boton1	Command1	Cancelar
Boton2	Command2	Suspender examen
Boton3	Command3	Confirmar
Boton4	Command4	Ayuda
Boton5	Command5	Pasaje
Boton6	Command6	>>
Boton7	Command7	<<
Temporizad	Timer1	
or		
Opción a	opA	
Opción b	opB	
Opción c	opC	
Opción d	opD	
Opción e	opE	
Opción f	opF (True)	

Acer.frm

	CtlName	Caption
Etiqueta1	LABEL1	Sistema Tutorial Versión 1.0
Etiqueta2	LABEL2	Universidad Nacional Autonoma de México
Etiqueta3	LABEL3	Tema de Tesis
Etiqueta4	LABEL4	Facultad de Ingeniería
Boton	COMMAND	Continuar
	1	
Imagen	OLE2	

Examen_x_Seccion.frm

	CtlName	Caption
Boton 1 al 10	Command1 .. Cmmand10	Nombre de las materias
Boton 11	Cmmand11	Cancelar

carga_calif.frm

	CtlName	Caption
Lista de Archivos	File1	
Marco	Frame3D1	
Etiqueta 1 a	Label1 ..	Datos del examen
Etiqueta 10	Label10	

carga_suspendidos.frm

	CtlName	Caption
Lista de Archivos	File1	
Marco	Frame3D1	
Etiqueta 1 a	Label1 ..	Datos del examen
Etiqueta 10	Label10	

Ejecutar.frm

	CtlName	Caption / Text
Etiqueta1	Label1	Proceso a ejecutar
Combinado1	Combo1	c:\
Mayúsculas	CheckBox	Mayúsculas
Boton1	Aceptar	Aceptar
Boton2	Cancelar	Cancelar

Salvar.frm

	CtlName	Caption/Text
Etiqueta 1 a	Label1 ..	Datos del examens a salvar
Etiqueta 5	Label5	Tipo de examen, etiquetas para las cajas de texto.
Texto 1 a	Text1 ..	Nombre, contraseña y verificación.
Texto 3	Text3	
Boton 1	Command1	Cancelar
Boton2	Command2	Aceptar

tipo_examen.frm

	CtlName	Caption
Boton1	Command1	Examen Global
Boton2	Command2	Examen por Sección

PanelCo.frm

	Caption	CtlName	Min	Max	SmallChange	LargeChange
Marco 1	Tamaño	Tamaño				
Etiqueta 1	Alto	Label1				
Etiqueta 2	Ancho	Label2				
Barra V.1		Alto	2500	10000	57	570
Barra V.2		Ancho	4000	10000	57	570
Marco 2	Color	Color				
Etiqueta 3	Rojo	Label3				
Etiqueta 4	Verde	Label4				
Etiqueta 5	Azul	Label5				
Barra H.1		Rojo	0	255	1	20
Barra H.2		Verde	0	255	1	20

Marco 3		Azul	0	255	1	20
Boton 1	Muestra	Muestra				
Boton 2	Aceptar	Aceptar				
Boton 3	Cancelar	Cancelar				

Ayud.frm

	CtlName	Caption
Caja Texto 1	Text1 ..	Nombres de las materias
a la	Text10	
Caja Texto 10		
Boton 1 a	Command 1	Nombres de los temas
Boton 72	..	
	Command 72	
Boton 73	Command 73	Salir

relo_digital.frm

	CtlName	Caption
Temporizad	Timer1	
or		
Hora	Text1	
Boton1	Command1	Quitar

Calculad.frm

	Caption	CtlName
Teclas 0 al 9	0,1,...,9	Dígito (arreglo de controles)
Teclas +,- ,x,/,*	+,-,x,/,*	Operación (arreglo de controles)
Punto decimal	.	PuntoD
Por ciento	%	TantoPorCiento
Cero Total	C	Inicializar
Cero corregir	CE	BorraEntrada

Preguntas_Global.frm

	ctlName	caption
Etiqueta 1	Label1	
Etiqueta 2	Label2	
Etiqueta 3	Label3	
Etiqueta 4	Label4	
Etiqueta 5	Label5	
Etiqueta 6	Label6	
Etiqueta 7	Label7	
Etiqueta 8	Label8	
Etiqueta 9	Label9	
Boton1	Command1	Cancelar
Boton2	Command2	Suspender examen
Boton3	Command3	Confirmar
Boton4	Command4	Ayuda
Boton5	Command5	Pasaje
Boton6	Command6	>>
Boton7	Command7	<<
Temporizador	Timer1	
Opción a	opA	
Opción b	opB	
Opción c	opC	
Opción d	opD	
Opción e	opE	
Opción f	opF (True)	
Lista 1 a lista 4	List1 .. list4	Datos de las preguntas. número, respuesta, materia
iconos	AniButton1 .. AniButton5	ayuda, reloj, pasaje, cancelar, lista.

Principal.frm		
	CtlName	Caption
Menú	Principal	
relo_analogico.frm		
	CtlName	Caption
Temporizad or	Timer1	
Revisa.frm		
	CtlName	Caption
Etiqueta 1	Label1	
Etiqueta 2	Label2	
Etiqueta 3	Label3	
Etiqueta 4	Label4	
Etiqueta 5	Label5	
Etiqueta 6	Label6	
Etiqueta 7	Label7	
Etiqueta 8	Label8	
Etiqueta 9	Label9	
Boton4	Command4	Ayuda
Boton5	Command5	Pasaje
Boton6	Command6	>>
Temporizador		
	Timer1	
Opción a	opA	
Opción b	opB	
Opción c	opC	
Opción d	opD	
Opción e	opE	
Opción f	opF (True)	
Boton 7	Command7	Salir

Preguntas_gauge.frm

	CtlName	Caption
Etiqueta 1	Label1	
Etiqueta 2	Label2	
Etiqueta 3	Label3	
Etiqueta 4	Label4	
Etiqueta 5	Label5	
Etiqueta 6	Label6	
Etiqueta 7	Label7	
Etiqueta 8	Label8	
Etiqueta 9	Label9	
Boton1	Command1	Cancelar
Boton2	Command2	Suspender examen
Boton3	Command3	Confirmar
Boton4	Command4	Ayuda
Boton5	Command5	Pasaje
Boton6	Command6	>>
Temporizador	Timer1	
Opción a	opA	
Opción b	opB	
Opción c	opC	
Opción d	opD	
Opción e	opE	
Opción f	opF (True)	
Medidor de respuestas	Gauge1	

Suspender_Tutor

	CtlName	Caption/Text
Boton 1	Command1	Aceptar
Boton 2	Command2	Cancelar

salvar_suspendido.frm

	CtlName	Caption/Text
Boton 1	Command1	Si
Boton 2	Command2	No
Boton 3	Command1	Cancelar

sal_suspendido.frm

	CtlName	Caption/Text
Etiqueta 1 a	Label1 ..	Datos del examens a salvar
Etiqueta 5	Label5	Tipo de examen, etiquetas para las cajas de texto.
Texto 1 a	Text1 ..	Nombre, contraseña y verificación.
Texto 3	Text3	
Boton 1	Command1	Cancelar
Boton2	Command2	Aceptar

Tutor_o_Examen

	CtlName	Caption/Text
Boton 1	Command1	Tutor
Boton 2	Command2	Examen

SELECCIÓN DEL LENGUAJE DE PROGRAMACIÓN.

El lenguaje que se utilizó para la programación del Sistema " Tutor " para la preparación del examen de selección al nivel medio superior, fué Visual BASIC Professional.

Aunque realmente la selección del lenguaje de programación no la realizamos durante esta etapa de Diseño, sino durante la última etapa de Análisis, es aquí donde documentaremos las causas por las cuales nos decidimos por Visual BASIC.

Introducción al lenguaje de programación Visual BASIC

La programación de Visual BASIC es una introducción práctica para la creación de aplicaciones Windows reales, con código BASIC; donde se crean aplicaciones gráficas con características de objetos, menús, ventanas redimensionales, soporte de ratón e iconos.

La programación en Visual BASIC difiere de la manera de programar tradicional; es decir, en lugar de crear un diagrama de flujo, donde se muestra el control del programa, la interrelación que existe de un procedimiento a otro y el flujo de la información dentro de los módulos del sistema.

En Visual BASIC se programa de manera "visual", esto es planear de modo que los controles puedan operar juntos para darle funcionalidad a cualquier aplicación Windows; su sistema de programación es modular, lo cual introduce un estilo efectivo y totalmente nuevo de programar en Windows.

Visual BASIC es una herramienta de desarrollo Windows, extraordinariamente poderosa y flexible, que le permite a los programadores crear aplicaciones dinámicas y sobre todo con rapidez; éste último fue uno de los principales criterios para la selección del lenguaje de programación.

Programación en Windows a la manera de Visual BASIC

El enfoque tradicional del desarrollo de una aplicación de Windows ha sido programar en "C" de Microsoft y utilizar el "Windows Software Development Kit". El ciclo de desarrollo transcurre de la siguiente manera: Se elabora un programa Windows esbozando los módulos principales y la lógica que los enlaza; después, se escribe el programa compilando, enlazando y depurando el código repetidamente. Los iconos, menús, cajas de diálogo y todos los controles en la interfaz del usuario se deben diseñar y compilar por separado; todos estos componentes deben ser controlados por el programa principal, a través de un laberinto de funciones y mensajes.

La filosofía de programación de Visual BASIC es, elaborar controles como ventanas, iconos, menús, etc.; después, escribir los procedimientos que invocan cada uno de los controles.

La interfaz Windows, es un ambiente operativo gráfico orientado visualmente para DOS, que permite mostrar y compartir datos entre varias aplicaciones al mismo tiempo; la mayor ventaja de utilizar Windows es que proporciona una interfaz consistente y manejable para desplegar información en muchas aplicaciones diferentes que comparten controles similares.

Programación de interfaces gráficas para el usuario a la manera de Visual BASIC

Una interfaz gráfica para el usuario GUI (Graphic user interface), es lo que el usuario observa cuando trabaja en una aplicación Windows. El utilizar una GUI permite al usuario interactuar con un programa y manipular sus características y funciones, además de proporcionar los controles que forman una interfaz del programa para ayudar al usuario a realizar sus tareas.

Visual BASIC permite diseñar la interfaz, escribir código en BASIC, y después asignar el código a los controles de la aplicación.

Con un intérprete como Visual BASIC, las instrucciones de los procedimientos pueden ejecutarse tan pronto como sean tecleadas. El

intérprete lee una línea de un procedimiento y envía las instrucciones apropiadas a la computadora para su ejecución; cuando la línea está completa, el intérprete se mueve a la siguiente línea, la interpreta y ejecuta.

Programas orientados a eventos

Una forma es una ventana que se crea y adapta a sus necesidades para cada aplicación, esto es la base de cualquier aplicación Visual BASIC que eventualmente correrá como un programa independiente en Windows. Un control es el nombre preciso para cualquier objeto que se dibuje en una forma; los botones de comando, menús, cajas de diálogo y barras de desplazamiento, son ejemplos de controles. Cuando un usuario activa un control al seleccionarlo, el código que está asignado al control es invocado y realiza una tarea específica. Un módulo es una estructura para escribir el código que se asigna a una forma y sus controles.

Orientado a eventos significa que todos los controles que se utilicen en una forma, especifican cómo se comportará la interfaz; en otras palabras, los controles de Visual BASIC esperan que sucedan eventos particulares para llevar a cabo una función o proceso. Un evento es una acción que es reconocida por un control.

Elementos de una aplicación de Visual BASIC

Control: Un término general utilizado para describir cualquier forma o elemento gráfico que se utilice dentro de una forma, incluyendo cajas de texto, cajas de listado, botones de comando, cajas de imágenes, barras de desplazamiento e íconos. Son los datos acoplados con una serie de rutinas o métodos que se utilizan exclusivamente para accesar y manipular el control.

Evento: La acción reconocida por un control de Visual BASIC.

Forma: Ventana que se crea y adapta a las necesidades de la aplicación a ser creada.

Método: Palabra de código de Visual BASIC que es similar a una función o instrucción, pero que siempre actúa sobre un control en particular; para cada control Visual BASIC predefine una serie de métodos que se pueden utilizar.

Procedimiento: Término que se refiere tanto a procedimientos **Sub** y **Function**. Es sencillamente una secuencia de instrucciones Visual BASIC que son ejecutadas en grupo durante la corrida.

Existen dos tipos de procedimientos: **Procedimientos de evento** que se limitan a las formas y los controles; y **Procedimientos generales** que se utilizan durante toda la aplicación y pueden ser requeridos por procedimientos de evento.

Proyecto: El conjunto de todos los archivos que forman la aplicación desarrollada.

Propiedad: Característica o atributo de un control. Para cada control Visual BASIC define una serie de propiedades que se aplican solamente a ese control.

Definición: Es el valor de una propiedad. Se puede modificar la definición de la mayoría de las propiedades mientras se esta construyendo una aplicación.

Diseño de una interfaz gráfica efectiva para el usuario

Una interfaz es aquel componente de una aplicación que traduce una acción del usuario en una o más peticiones para desarrollar funciones y retroalimentar al usuario con las consecuencias y sus acciones.

La creación de una aplicación efectiva comienza con un buen diseño y una visión amplia del contenido y propósito de la aplicación; la manera en que se organiza una aplicación depende de la información que ésta contiene y de cómo se espera que el usuario la utilice.

Se deberá tener una aplicación en la cual los usuarios navegarán fácilmente en ella siguiendo una trayectoria lógica, intuitiva y no lineal marcada con indicaciones claras. Una aplicación bien diseñada debe guiar a los usuarios de un punto a otro, permitiéndole abandonar el proceso en cualquier momento, por lo tanto, los controles que estén dentro de la aplicación se convertirán en las señales que guiarán al usuario dentro de la aplicación.

Control de una aplicación

La manera en la cual se le ofrece a los usuarios la sensación de controlar una aplicación en Windows es hacer que ellos la manipulen directamente, como al abrir y cerrar una aplicación; el transitar dentro de una aplicación; el escoger objetos; el llevar a cabo un evento de una serie de controles; etc. Los usuarios deben esperar que sus acciones físicas obtengan resultados físicos.

Ej. Cuando los usuarios escogen un menú, aparece una lista de comandos; cuando hace clic sobre un botón de minimización, se encoge una ventana; cuando hace clic sobre un texto, se puede realzar y cambiar sus atributos. Estos son resultados reales de sus acciones, los usuarios deben obtener una retroalimentación en tiempo de ejecución inmediata. Esto ayudará a reforzar continuamente la utilización de una aplicación.

Al mismo tiempo, Visual BASIC da los recursos para crear aplicaciones que presenten un ambiente visualmente dinámico para que los usuarios completen sus tareas, en donde puedan seguir rutas no lineales proporcionadas por objetos. Los usuarios pueden interactuar de manera directa con la pantalla, escogiendo objetos y tareas que a ellos les interesa explorar con sólo señalarlos.[SAN91]

La selección del lenguaje de programación la realizamos durante la fase de análisis en la etapa de especificación del sistema y es en esta sección donde documentamos lo concerniente a dicha selección.

Los principales criterios por los cuales nos guiamos para seleccionar VisualBASIC fueron: el poder manejar el ambiente windows, el costo del paquete, la disponibilidad de la licencia para trabajar con él, aprovechar el

sistema de mensajes de windows que es el "corazón" de la programación orientada a eventos; la disponibilidad de controles ya programados , disponibles para utilizarse, cada uno con propiedades establecidas y metodos aplicables, con la ventaja de poder asociar código (de un lenguaje procedural como lo es el BASIC de Micro-Soft) a cada uno de dichos controles.

IV. DESARROLLO

IV.1 CÓDIGO

TECNICAS DE PROGRAMACION Y CODIFICACION

El uso de técnicas de programación se enfoca hacia lo que es la fase de desarrollo y el diseño del software. La fase de desarrollo comprende tres pasos distintos: diseño, generación del código y prueba. Cada paso transforma la información de forma que finalmente se obtiene un software de computadora validado.

Proceso de diseño: El diseño de programas es un proceso mediante el que se traducen los requerimientos en una representación del software. Desde el punto de vista de una gestión de proyecto, el diseño de software se realiza en dos pasos. El diseño preliminar se refiere a la transformación de los requerimientos en datos y arquitectura del software. El diseño en detalle se enfoca hacia los refinamientos de la representación arquitectónica que conduce a una estructura de datos detallada y a representaciones algorítmicas del software.

Fundamentos del diseño: En las últimas tres décadas se ha establecido un conjunto de conceptos fundamentales en el diseño de software. ¿ Qué criterios pueden usarse para subdividir el software en componentes individuales ?, ¿ Cómo se separan los detalles de una función o estructura de datos de una representación conceptual del software ?, ¿ Existen criterios uniformes que definen la calidad técnica de un diseño de programas ?.

“El comienzo del saber de un programador de computadoras es la diferencia entre obtener un programa que funciona y obtener uno que funciona correctamente”.

Refinamiento: El refinamiento sucesivo es una temprana estrategia de diseño descendente. La estructura de programa se desarrolla en niveles sucesivos de refinamiento de los detalles procedimentales. En cada paso (del refinamiento), una o varias instrucciones del programa dado se descomponen

en más instrucciones detalladas. Esta descomposición sucesiva o refinamiento de especificaciones termina cuando todas las instrucciones están expresadas en términos del lenguaje de programación. Conforme se refinan las tareas, también los datos pueden ser refinados, descompuestos o estructurados. Es normal refinar el programa y las especificaciones de los datos en paralelo.

Estructura del programa: La estructura del programa representa la organización (frecuentemente jerárquica) de los componentes del programa (módulos) e implica una jerarquía de control. No representa aspectos procedimentales del software, tales como la secuencia de procesos, ocurrencia, orden de decisiones o repetición de operaciones.

Para representar una estructura de programa se utilizan muchas notaciones diferentes. La más común es un diagrama de árbol frecuentemente llamado de estructura. La estructura del programa define la jerarquía de control independientemente de las decisiones y secuencias de procesamiento.

Modularidad: Se ha establecido que la "modularidad es un atributo del software que permite a un programa ser independiente". El software monolítico (es decir, un gran programa compuesto de un único módulo) no puede ser fácilmente abarcado por un lector. El número de caminos de control, expansión de referencias, número de variables y complejidad global podría hacer imposible el comprenderlo bien.

La modularidad se ha convertido en un enfoque acertado. Un diseño modular, reduce la complejidad, facilita los cambios (un aspecto crítico del mantenimiento) y da como resultado una implementación más fácil posibilitando el desarrollo paralelo de diferentes partes de un sistema.

Diseño arquitectónico

El objetivo principal del diseño arquitectónico es desarrollar una estructura de programa modular y representar las relaciones de control entre los módulos. Además, el diseño arquitectónico mezcla la estructura de programas, la estructura de datos y define las interfaces que facilitan el flujo de los datos a lo largo del programa.

Diseño procedimental

El diseño procedimental se realiza después de que se ha establecido la estructura del programa y de los datos. La especificación procedimental requiere definir los detalles algorítmicos que deben establecerse en un lenguaje natural. Después de todo, los miembros de un equipo de desarrollo de software hablan un lenguaje natural (en teoría, al menos), la gente exterior al dominio del software comprendería más fácilmente la especificación y no se requeriría un nuevo aprendizaje.

Desafortunadamente, existe un pequeño problema. El diseño procedimental debe especificar los detalles de los procedimientos sin ambigüedad y la falta de ambigüedad en un lenguaje natural no es corriente. Usando un lenguaje natural, podemos escribir un conjunto de pasos procedimentales en demasiadas formas diferentes. Frecuentemente, un significado concreto depende del contexto.

Programación estructurada: Los fundamentos del diseño procedimental se formaron a principios de los años 60's y se solidificaron con el trabajo de Edsger Dijkstra y sus colegas. A finales de la misma década, Dijkstra y otros propusieron el uso de un conjunto de construcciones lógicas con las que podría formarse cualquier programa. Las construcciones reforzaban el "mantenimiento del dominio funcional". Esto es, cada construcción tenía una estructura lógica predecible, se entraba a ella por el principio y se salía por el final y facilitaba a el lector seguir más fácilmente el flujo procedimental.

La secuencia implementa los pasos de procesamiento esenciales en la especificación de cualquier algoritmo, la condición da la facilidad para seleccionar un procedimiento basado en alguna ocurrencia lógica y la repetición suministra el bucle. Estas tres construcciones son fundamentales en programación estructurada.

Las estructuras de control se propusieron para limitar el diseño procedimental del software a un pequeño número de operaciones predecibles. El uso de construcciones estructuradas reduce la complejidad de los programas y por lo tanto facilita la legibilidad, prueba y mantenimiento. El uso de un número limitado de construcciones lógicas también contribuye a

un proceso de comprensión humana que los psicólogos llaman troceo. Las construcciones estructuradas son trozos lógicos que permiten a un lector reconocer los elementos procedimentales de un módulo en vez de leer el diseño o código línea a línea. La comprensión aumenta cuando se encuentran formas lógicas fácilmente reconocibles.

Cualquier programa, independientemente del área de aplicación o complejidad técnica, puede diseñarse e implementarse usando sólo las tres construcciones estructuradas.

Técnicas de descomposición

Los seres humanos hemos desarrollado una aproximación natural a la resolución de problemas: si el problema a resolver es demasiado complicado, tendemos a subdividirlo hasta encontrar problemas manejables. Entonces resolvemos cada uno individualmente y esperamos que las soluciones puedan ser combinadas para responder al problema original.

La estimación del proyecto de software es una forma de resolución de problemas y, en la mayoría de los casos, el problema a resolver (o sea, desarrollar estimaciones de costos y esfuerzo para un proyecto de software) es demasiado complejo para considerarlo como una sola pieza. Por esta razón, descomponemos el problema, re-caracterizándolo como un conjunto de pequeños problemas (esperando que sean más manejables).

Técnicas de cuarta generación

El término "técnicas de la cuarta generación" (T4G) abarca un amplio espectro de herramientas de software que tienen una cosa en común: todas facilitan el desarrollo del software. La herramienta genera automáticamente el código fuente basándose en la especificación del técnico. Existe cierto debate sobre cuánto puede aumentarse la rapidez en la construcción de un programa respecto a cuánto a de levantarse el nivel en el que se especifique el software para una máquina. El paradigma T4G para la ingeniería del software se orienta hacia la habilidad de especificar software a un nivel que sea más

próximo al lenguaje natural o en una notación que proporcione funciones significativas.

Actualmente, un entorno para el desarrollo del software que soporte el paradigma T4G incluye algunas o todas de las siguientes herramientas: lenguajes no procedimentales para consulta a bases de datos, generación de informes, manipulación de datos, interacción y definición de pantallas y generación de código; capacidades gráficas de alto nivel; y capacidad de hoja de cálculo. Cada una de estas herramientas existen, pero solo para dominios de aplicación muy específicos. No existe hoy disponible un entorno T4G que pueda aplicarse con igual facilidad a todas las categorías de aplicaciones del software.

IV.2 TÉCNICAS DE PRUEBA DEL SOFTWARE

La prueba de software es un elemento crítico para la garantía de calidad del software y representa un último repaso de las especificaciones, del diseño y de la codificación .

El diseño de casos de prueba se centra en un conjunto de técnicas para la creación de casos prueba que satisfagan los objetivos globales de la prueba.

OBJETIVOS DE LA PRUEBA

El objetivo es diseñar pruebas que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo. Como ventaja secundaria la prueba demuestra hasta qué punto las funciones del software satisfacen las especificaciones y parecen alcanzarse los requerimientos de rendimiento. Además, los datos que se van recogiendo a medida que se lleva a cabo la prueba proporcionan una buena indicación de la fiabilidad del software y de alguna manera indican la calidad del software como un todo.

Al llevarse a cabo la prueba se evalúan los resultados; es decir, se comparan los resultados de la prueba con los esperados. Cuando se descubren datos

erróneos, esto implica que hay un error y comienza la depuración. El proceso de depuración es una consecuencia impredecible de la prueba.

Al ser recopilados y evaluados los resultados de la prueba comienza a vislumbrarse una medida cualitativa de la calidad y la fiabilidad del software.

DISEÑO DE CASOS DE PRUEBA

Cualquier producto de ingeniería (y muchas otras cosas) puede ser probado de dos formas:

1) Conociendo la función específica para la que fue diseñado el producto, se pueden llevar a cabo pruebas que demuestren que cada función es completamente operativa.

2) Conociendo el funcionamiento del producto, se pueden desarrollar pruebas que aseguren que "todas las piezas encajan"; o sea, que la operación interna se ajusta a las especificaciones y que todos los componentes internos se han comprobado de forma adecuada.

La primera aproximación de prueba se denomina prueba de la caja negra y la segunda prueba de la caja blanca.

Los atributos de la prueba de la caja blanca así como de la caja negra se pueden combinar para llegar a una aproximación que valide la interfaz del software y asegure selectivamente que el funcionamiento interno del software es correcto.

Prueba de la Caja Blanca

La prueba de la caja blanca es un método de diseño de casos de prueba que usa la estructura de control del diseño procedimental para derivar los casos de prueba. Mediante los métodos de prueba de la caja blanca el ingeniero del software puede derivar casos de prueba que:

- 1) Garanticen que se ejercitan por lo menos una vez todos los caminos independientes de cada módulo.
- 2) Se ejercitan todas las decisiones lógicas en sus opciones, verdadera y falsa.
- 3) Se ejecutan todos los bucles en sus límites y con sus límites operacionales
- 4) Se ejercitan las estructuras de datos internas para asegurar su validez.

Prueba del Camino Básico

El método del camino básico permite al diseñador de casos de prueba derivar una medida de complejidad lógica de un diseño procedural y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución. Los casos de prueba derivados del conjunto básico garantizan que durante la prueba se ejecuta por lo menos una vez cada sentencia del programa.

Prueba de la Caja Negra

Los métodos de prueba de la caja negra se centran en los requerimientos funcionales del software, es decir, la prueba de la caja negra permite al ingeniero del software derivar conjuntos de condiciones de entrada que ejerciten completamente todos los requerimientos funcionales de un programa.

La prueba de la caja negra intenta encontrar errores de las siguientes categorías:

- 1) Funciones incorrectas o ausentes.
- 2) Errores de interfaz.

3) Errores en estructuras de datos o en accesos a bases de datos externas.

4) Errores de rendimiento.

5) Errores de inicialización y de terminación.

A diferencia de la prueba de la caja blanca, que se lleva a cabo previamente en el proceso de prueba, la prueba de la caja negra tiende a ser aplicada durante posteriores fases de prueba. Ya que la prueba de la caja negra intencionadamente ignora la estructura de control, concentra su atención en el dominio de información.

Partición equivalente

La partición equivalente es un método de prueba de la caja negra que divide el dominio de entrada de un programa, en clases de datos de los que se pueden derivar casos de prueba. El diseño de casos de prueba para la partición equivalente se basa en una evaluación de las clases de equivalencia para una condición de entrada. Una clase de equivalencia representa un conjunto de estados válidos o inválidos para condiciones de entrada. Típicamente una condición de entrada es un valor numérico específico, un rango de valores, un conjunto de valores relacionados o una condición Booleana (sí o no). Las clases de equivalencias se pueden definir de acuerdo con las siguientes directrices:

1. Si una condición de entrada especifica un rango, se definen una clase de equivalencia válida y dos inválidas.

2. Si una condición de entrada requiere un valor específico, se definen una clase de equivalencia válida y dos inválidas.

3. Si una condición de entrada especifica un miembro de un conjunto, se definen una clase de equivalencia válida y una inválida.

4. Si una condición de entrada es Booleana, se definen una clase válida y una inválida.

Análisis de valores límite

Por razones que no están del todo claras, los errores tienden a darse más en los límites del dominio de entrada que en el "centro". Es por esto que se ha desarrollado el análisis de valores límites (AVL) como técnica de prueba.

El análisis de valores límite es una técnica de diseño de casos de prueba que complementa a la partición equivalente. En lugar de seleccionar cualquier elemento de una clase de equivalencia, el AVL lleva a la elección de casos de prueba en los "bordes" de la clase. En lugar de centrarse solamente en las condiciones de entrada, el AVL deriva casos de prueba también para el dominio de salida [MYE79].

Las directrices de AVL son similares en muchos aspectos a las que proporciona la partición equivalente:

1. Si una condición de entrada especifica un rango limitado por los valores a y b, se deben diseñar casos de prueba para los valores a y b y para los valores justo por debajo y justo por encima de a y b, respectivamente.
2. Si una condición de entrada especifica un número de valores, se deben desarrollar casos de prueba que ejerciten los valores máximo y mínimo. También se deben probar los valores justo por encima y justo por debajo del máximo y del mínimo.
3. Aplicar las directrices 1 y 2 a las condiciones de salida.

Pruebas de Validación de Datos

1. Especificar órdenes con sintaxis incorrecta; utilizar versiones de la forma correcta tanto evidentes como sutiles.

2. Proporcionar entradas sintácticamente correctas, pero que se encuentren fuera de secuencia o que estén especificadas en un momento incorrecto.
3. Escribir una orden parcialmente correcta y a continuación terminar la entrada de órdenes.
4. Omitir todas las ordenes; simplemente pulsar "Return".
5. Proporcionar ordenes correctas, sin muchos datos de parámetros.

La partición equivalente y el análisis de valores límite son técnicas de prueba complementarias importantes para la validación

IV.3 ESTRATEGIAS DE PRUEBA

Una estrategia de prueba de software integra las técnicas de diseño de casos de prueba en una serie de pasos bien planificados que llevan a una construcción correcta del software. Cualquier estrategia de prueba debe incorporar la planificación de la prueba, el diseño de casos de prueba, la ejecución de pruebas, la recolección y evaluación de los datos resultantes.

Una estrategia para el software debe acomodar pruebas de bajo nivel que verifiquen que cada pequeño segmento de código fuente ha sido implementado correctamente, así como pruebas de alto nivel que muestren la validez de las principales funciones del sistema frente a los requerimientos del cliente.

Verificación y validación

La verificación se refiere al conjunto de actividades que aseguran que el software implementa correctamente una función específica. La validación se refiere a un conjunto diferente de actividades que aseguran que el software construido se ajusta a los requerimientos del cliente.

La prueba es una serie de cuatro pasos que se llevan a cabo secuencialmente:

- Prueba de Unidad**
- Prueba de Integración**
- Prueba de Validación**
- Prueba del Sistema**

Prueba de Unidad

La prueba de unidad centra el proceso de verificación en la menor unidad del diseño del software, el módulo. Usando la descripción del diseño detallado como guía, se prueban los caminos de control importantes con el fin de descubrir errores dentro del ámbito del módulo. La prueba de unidad siempre está orientada a la caja blanca, y este paso se puede llevar a cabo en paralelo para múltiples módulos.

Se prueba la interfaz del módulo para asegurar que la información fluye de forma adecuada hacia y desde la unidad del programa que está siendo probada. Se examinan las estructuras de datos locales para asegurar que los datos que se mantienen temporalmente conservan su integridad durante todos los pasos de ejecución del algoritmo. Se prueban las condiciones límite para asegurar que el módulo funciona correctamente en los límites establecidos como restricciones de procesamiento. Se ejercitan todos los caminos independientes (caminos básicos) de la estructura de control con el fin de asegurar que todas las sentencias del módulo se ejecutan por lo menos una vez. Y finalmente, se prueban todos los caminos de manejo de errores.

Además de las estructuras de datos locales, durante la prueba de unidad se debe comprobar (en la medida de lo posible) el impacto de los datos globales sobre el módulo.

Prueba de Integración

La prueba de integración es una técnica sistemática para construir la estructura del programa mientras que al mismo tiempo se llevan a cabo pruebas para detectar errores asociados con la interacción. El objetivo es coger los módulos probados en unidad y construir una estructura de programa que esté de acuerdo con lo que dicta el diseño.

El programa se construye y se prueba en pequeños segmentos en los que los errores son más fáciles de aislar y de corregir, es más probable que se puedan probar completamente las interfaces y se puede aplicar una aproximación de prueba sistemática.

Integración descendente

La integración descendente es una aproximación incremental a la construcción de la estructura de programas. Se integran los módulos moviéndose hacia abajo por la jerarquía de control, comenzando con el módulo de control principal (programa principal). Los módulos subordinados al módulo de control principal se van incorporando en la estructura.

El proceso de integración se lleva a cabo en una serie de cinco pasos:

1. Se usa el módulo de control principal como conductor de la prueba, disponiendo resguardos para todos los módulos directamente subordinados al módulo de control principal.
2. Dependiendo de la aproximación de integración elegida (o sea, primero-en-profundidad o primero-en-anchura) se van sustituyendo los respaldos subordinados uno a uno por los módulos reales.
3. Se llevan a cabo pruebas cada vez que se integra un nuevo módulo.
4. Tras terminar cada conjunto de pruebas, se reemplaza otro respaldo con el módulo real.
5. Se hace la prueba de regresión (o sea, todas o algunas de las pruebas anteriores) para asegurar que no se han introducido nuevos errores.

El proceso continúa desde el paso 2 hasta que se haya construido la estructura del programa entero.

La estrategia de integración descendente verifica los puntos de decisión o de control principales más pronto en el proceso de prueba. En una estructura de programa bien fabricada, la toma de decisiones se da en los niveles superiores de la jerarquía y por tanto se encuentran antes.

En todas las fases de prueba se siguen los siguientes criterios con sus correspondientes pruebas:

Integridad de interfaz. Se prueban las interfaces internas y externas a medida que se incorpora cada módulo (o grupo) a la estructura.

Validez funcional. Se llevan a cabo pruebas diseñadas para descubrir errores funcionales.

Contención de información. Se llevan a cabo pruebas diseñadas para descubrir errores asociados con las estructuras de datos globales o locales.

Rendimiento. Se llevan a cabo pruebas diseñadas para verificar los límites de rendimiento establecidos durante el diseño del software. Para el caso del sistema " Tutor " no se justifica; es decir " Resultaría más caro el aire que las llantas ".

Prueba de Validación

Tras la culminación de la prueba de integración, el software está completamente ensamblado como un paquete; se han encontrado y corregido los errores de interfaces, y debe comenzar una serie final de pruebas del software lo que conocemos como prueba de validación. La validación puede ser definida de muchas formas, pero una simple (si bien ruda) indicación es que la validación se logra cuando el software funciona de acuerdo con las expectativas razonables del cliente.

La validación del software se consigue mediante una serie de pruebas de la caja negra que demuestran la conformidad con los requerimientos.

Pruebas alfa y beta

La prueba alfa es conducida por un cliente (*alpha-tester*) en el lugar de desarrollo. Se usa el software de forma natural, con el encargado del desarrollo "mirando por encima del hombro" del usuario y registrando errores y problemas de uso. Las pruebas alfa se llevan a cabo en un entorno controlado.

La prueba beta se lleva a cabo en uno o más lugares de clientes (*beta-testers*) por los usuarios finales del software. A diferencia de la prueba alfa, el encargado del desarrollo normalmente no está presente. Así, la prueba beta es una aplicación "en vivo" del software en un entorno que no puede ser controlado por el equipo de desarrollo. El cliente registra todos los problemas (reales o imaginarios) que encuentra durante la prueba beta e informa a intervalos regulares al equipo de desarrollo. Como resultado de los problemas anotados durante la prueba beta, el equipo de desarrollo del

software lleva a cabo modificaciones y así prepara una versión del producto de software para toda la base de clientes.

Prueba del Sistema

El software una vez validado, se debe combinar con otros elementos del sistema (hardware, gente, bases de datos), la prueba del sistema verifica que cada elemento se acople adecuadamente, alcanzando la funcionalidad y el rendimiento del sistema total.

La prueba del sistema realmente está constituida por una serie de pruebas diferentes cuyo propósito primordial es ejercitar profundamente el sistema basado en computadora. Aunque cada prueba tiene un propósito distinto, todas trabajan para verificar que se han integrado adecuadamente todos los elementos del sistema y que realizan las funciones apropiadas.

Algunas de las pruebas son las siguientes:

- Pruebas de recuperación
- Pruebas de seguridad
- Pruebas de resistencia
- Pruebas de rendimiento

El proceso de depuración

La depuración aparece como una consecuencia de una prueba efectiva. O sea, cuando un caso de prueba descubre un error, la depuración es el proceso que resulta en la eliminación del error.

La depuración comienza con la ejecución de un caso de prueba. Se evalúan los resultados y aparece una falta de correspondencia entre los esperados y los reales. En muchos casos, los datos que no corresponden son un síntoma de una causa subyacente que todavía permanece oculta. El proceso de depuración intenta hacer corresponder el sistema con una causa, llevando así a la corrección del error.

Independientemente del enfoque que se utilice, la depuración tiene un objetivo primordial: encontrar y corregir la causa de un error en el software. El objetivo se consigue mediante una combinación de una evaluación sistemática y de una gran cantidad de intuición e inclusive de suerte.

La vuelta atrás es un enfoque más normal para la depuración que se puede usar con éxito para pequeños programas. Partiendo del lugar donde se descubre el síntoma, se recorre hacia atrás (manualmente) el código fuente hasta que se llega a la posición del error. Desgraciadamente, a medida que aumenta el número de líneas del código, el número de posibles caminos de vuelta se hace inmanejablemente grande.[PRE88]

IV.4 PRUEBAS DEL SISTEMA "TUTOR"

Pruebas del sistema

Basándonos en el marco teórico descrito anteriormente y en nuestra experiencia como programadores, se tomaron las siguientes consideraciones. Conforme a :

- Las características, especificaciones, requerimientos, funciones, alcances y objetivos del sistema.
- Las características propias del lenguaje en el cual se programó el sistema "Tutor" (Visual Basic) y el ambiente operativo (Windows).

Se llevó a cabo una estrategia de prueba del sistema, involucrando las técnicas de prueba que consideramos pertinentes, para garantizar ante todo la calidad, el buen funcionamiento y eficiencia del Sistema " Tutor ".

Dentro de la estrategia de prueba se cubrieron aspectos de gran importancia para poder brindar al usuario un producto atractivo, eficiente y al mismo tiempo cumpliendo con los objetivos planteados en la etapa de desarrollo.

La estrategia de prueba constó de las siguientes partes:

1) Verificar y validar la Unidad e Integración del Sistema " Tutor ", utilizando las técnicas de prueba en la etapa de desarrollo del sistema.

2) Validación y verificación de los conjuntos de información empleados en el sistema " Tutor ".

3) Validación del Sistema " Tutor " como producto terminado.

1.Verificar y validar la Unidad e Integración del Sistema " Tutor ".

La manera de programar en Visual Basic es "visual"(valga la redundancia); esto es, se planea de manera tal que los controles involucrados en los diferentes módulos del sistema, puedan operar juntos para darle funcionalidad a cualquier aplicación windows. Además, su programación es modular orientada a eventos; esto es, se programa cada módulo de manera independiente, vinculando por medio de los diferentes eventos los conjuntos de información del sistema y/o módulos correspondientes.

De tal manera que al estar elaborando el sistema se hizo uso de la técnica de prueba de la caja blanca, para verificar las unidades (módulos) que componen el sistema; al mismo tiempo comprobar los diferentes caminos de control de cada uno de ellos, verificando su integridad.

Esta prueba la realizamos durante la etapa que Visual Basic denomina tiempo de diseño; es decir, se lleva a cabo previamente en el proceso de prueba a diferencia de la prueba de la caja negra que tiende a ser aplicada en tiempo de ejecución.

Se probó la ejecución de cada unidad de forma independiente para verificar la interfaz del módulo asegurando que el flujo de la información fuera el adecuado; probando las estructuras de datos locales y visualizando la integridad de los mismos, durante todos los pasos de ejecución del algoritmo empleado en la unidad. Además, al estar creando los controles utilizados en cada unidad se verificó tanto las respuestas esperadas al evento asignado como los conjuntos de información asociados al código de cada forma.

Partiendo de la carta de navegación planteada para la realización del sistema, se llevó a cabo la prueba del camino básico, ejecutando cada uno de los eventos asociados a los módulos involucrados en el sistema para garantizar su operatividad.

Realizamos la prueba del "camino básico" para forzar la ejecución de cada una de las líneas de código del programa.

2. Validación y verificación de los conjuntos de información empleados en el sistema "Tutor"

En el caso de los hipertextos realizamos pruebas de tipo exhaustivo. Para la validación de las bases de datos, realizamos modificaciones a los algoritmos que muestran las preguntas de una manera aleatoria a una manera secuencial, realizando de igual modo pruebas de tipo exhaustivo. También realizamos la prueba de los valores límite, además contamos con la validación de expertos en la materia para comprobar la integridad de la información.

3. Validación del sistema "Tutor" como producto terminado

Realizando pruebas de bajo nivel y alto nivel, para así verificar que cada pequeño segmento de código se implementó correctamente; y comprobar que las funciones y procesos cumplan con los objetivos planteados para la realización del mismo.

Utilizamos cargas de trabajo reales las cuales arrojaron los ajustes pertinentes, funciones ausentes, errores en estructura de datos, en acceso de bases de datos externas, errores de inicialización y terminación y sobre todo errores de interfaz.

Realizamos pruebas de tipo alfa y beta .

CONCLUSIONES

CONCLUSIONES

Habiendo encausado nuestros conocimientos y vivencias personales de acuerdo con los objetivos planteados y nuestra propuesta del método de solución, llegamos a las siguientes conclusiones:

Durante el desarrollo de este trabajo de Tesis, en el aspecto personal adquirimos gran experiencia en cuestión de coordinar el esfuerzo y conocimiento de cada uno para la culminación de un objetivo en común; responsabilizándonos al repartir, aceptar, y realizar cada una de las tareas asignadas siendo la comunicación, confianza y disciplina el estandarte del equipo para la realización de este trabajo en singular armonía.

Con este trabajo, aportamos una solución adecuada para la preparación del examen de admisión al ciclo de bachillerato y consideramos que este sistema tomará un papel de punta de lanza para abrir el camino en el desarrollo de sistemas similares.

Con el diseño realizado por el equipo, llegamos a la conclusión de que esta filosofía es una nueva técnica de estudio introducida, para ingreso a ciclos de bachillerato, con una gran riqueza en conocimiento y en definitiva con la alternativa de aumentar aun más esta fuente. El uso del sistema cambiaría totalmente el ambiente de estudio y aumentaría la capacidad de retención precisamente por el enfoque que se pretendió darle. Inclusive, un sistema de este estilo no solo está destinado para alumnos que tienen que presentar exámenes de admisión, sino que además sirve como una herramienta de enseñanza muy interesante para quienes deseen aumentar su cultura general.

En cuanto a quienes estaría dirigido el sistema " Tutor " y quienes puedan adquirirlo, concluimos que en caso de comercializar el sistema " Tutor " sería fácilmente adquirible para aquellas personas que hacen uso de computadoras personales.

Referente a costos por diseño y desarrollo después del análisis realizado, concluimos que esta herramienta resultaría costeable llevando a cabo una adecuada introducción al mercado. Por las características que posee, por no

tener un límite en cuanto a nuevos alcances, nuevas áreas de aplicación, por ser nuevo en su tipo y a quienes estará destinado; el sistema " Tutor ", en definitiva ofrece agradables expectativas.

Al aprender a realizar una aplicación Windows, ubicamos el alcance y grado de complejidad del lenguaje Visual BASIC y del compilador HC31 para realizar ayuda relacionada al contexto; con lo cual consideramos que este tipo de programación en el presente tiene gran impacto y es una tendencia a futuro, estamos realizando programación windows en un ambiente que es ampliamente aceptado en el mercado y que seguramente lo será a futuro.

El desarrollo de este trabajo de tesis, nos ha permitido vincular los conocimientos adquiridos en la Universidad, trabajo y experiencias personales; es decir, hemos aprovechado la oportunidad de aplicar parte de los conocimientos adquiridos en nuestra formación académica, a una situación real, proponiendo para un problema real una solución adecuada.

Con la creación de este sistema estamos atacando un punto de gran relevancia para nuestro país y da la pauta para el desarrollo de sistemas educativos similares exhortando a los alumnos, profesores, investigadores y autoridades de la Facultad de Ingeniería a coordinar sus esfuerzos para llevarlos a cabo.

Cuando se planteó la propuesta del desarrollo del sistema " Tutor ", se hizo un estudio detallado de los alcances y objetivos que se pretendían realizar. Después de la etapa de diseño y desarrollo que se han efectuado, podemos darnos cuenta de la cantidad de herramientas que pueden realizarse con el mismo enfoque.

Recomendamos ampliamente a los estudiantes de secundaria se acerquen a utilizar esta herramienta con la que pueden disponer de una serie de elementos que conllevan a un solo camino: **un alto rendimiento**. Con lo que podemos concluir que los resultados esperados han sido superados satisfactoriamente.

Hemos asimilado la importancia y el impacto que un sistema de esta naturaleza puede ocasionar en un sector estudiantil; además de contar con

una respuesta inesperada, ya que gente adulta de diferente edad, opina en general que éste es un sistema muy bueno de cultura general e inclusive de consulta para temas específicos.

EXPECTATIVAS Y UN VISTAZO AL FUTURO

Tomando conciencia de la importancia y responsabilidad que conlleva el desarrollo de un sistema de esta naturaleza, es importante mencionar que como desarrolladores del sistema "Tutor", proponemos que en un futuro cuente con un módulo diseñado para el profesorado, interesado en incrementar el número de preguntas o adecuarlas a los planes de estudio vigentes, haciendo más flexibles y efectivos los conjuntos de información de entrada del sistema, es decir, las bases de datos; otro módulo destinado a la elaboración de exámenes "tipo", diseñados a la medida y otro módulo el cual permita responder en cada una de las opciones múltiples, el porque una respuesta es correcta y cada una de las restantes no lo son.

En el caso de comercializar el sistema "Tutor" en los países de habla hispana (Latinoamérica, España, Filipinas) debido a al globalización económica mundial y a nuestra privilegiada ubicación geográfica, puede colocarse en el mercado de software de dichos países y por la naturaleza del mismo, esperamos que sea con gran aceptación.

El TLC (Tratado de Libre Comercio) hace vislumbrar que en un futuro no muy lejano se deban de crear estándares de planes de estudio en los tres países (EE.UU, Canadá y México), los cuales garanticen un nivel mínimo de conocimientos en todos los niveles educativos, de aquí que el desarrollo de software diseñado a la preparación de resolución de exámenes tiene un gran futuro.

Pronosticamos que los sistemas tutoriales en computadora como herramienta complementaria para el estudio, tomarán un papel de tipo doméstico en un corto plazo.

Finalmente, esperamos que el sistema "Tutor" sea un eslabón más en la interminable cadena del saber.

BIBLIOGRAFÍA

BIBLIOGRAFIA

- [WB94] Diccionario Webster 1994.
- [SAN91] Donald H. Sanders, Informática Presente y Futuro 1991, McGraw-Hill.
- [DEI89] Harvey M. Deitel, Introducción a los Sistemas Operativos 1989, Addison Wesley.
- [WID90] Gio Widerhold, Bases de Datos 1990, Addison Wesley.
- [IE85] Instituto de Investigaciones Eléctricas, Desarrollo y Administración de Programas de Computadora (Software) 1985, CECSA.
- [PCM94] Personal Computer México, Vol. Agosto 1994.
- [PRE88] Roger S. Pressman, Ingeniería del Software Un Enfoque Práctico 1988, McGraw-Hill.
- [ISH86] Dr. Ishikawa, Control Total de Calidad 1986.
- [SCH92] Guadalupe Morales G., Guía para Ingresar a Preparatoria y CCH 1992, Serie Schaum McGraw-Hill.
- [MYE79] Myes G., The Art of Software Testing 1979, Wiley.
- [VIS93] Emilio L. Sanchis Pastor, Visual Basic Guía Práctica para programadores 1993, Anaya.
- [MVB92] Fco. Javier Ceballos, MS Visual Basic Aplicaciones para Windows 1992, Ra-ma.
- [FEL93] Phil Feldman, Using Visual Basic 3 1993, Que.

APENDICE "A"
CODIGO

1. 1940

2. 1941

3. 1942

4. 1943

5. 1944

6. 1945

7. 1946

8. 1947

9. 1948

10. 1949

11. 1950

```

Sub Command3D1_Click ()
Materia = "Biologia"
TipoExamen = -1
Num_pregunta = 1
numero_maximo_de_registros = 202
fin = Timer + 1200 'intervalo del examen en segundos
Set MiBase = OpenDatabase("c:\paquet\dbase\","c:\paquet\dbase\","dbase IV;")
Set MiTabla = MiBase.OpenTable(Materia)
MiTabla.MoveFirst
num_aleatorio = aleatorio(numero_maximo_de_registros)
aleat_anterior = num_aleatorio
For j = 1 To num_aleatorio - 1
    MiTabla.MoveNext
Next j
Respuesta(1).NoPreg = num_aleatorio
Respuesta(1).R = "F" sin contestar
Respuesta(1).N = MiTabla!Respuesta
If MiTabla!Pregunta <> nul Then
    preguntas_x_seccion.Label1.Caption = MiTabla!Pregunta
Else
    preguntas_x_seccion.Label1.Caption = ""
End If
If MiTabla!A <> nul Then
    preguntas_x_seccion.opA = True
    preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
Else
    preguntas_x_seccion.opA = False
    preguntas_x_seccion.Label2.Caption = ""
End If
If MiTabla!B <> nul Then
    preguntas_x_seccion.opB = True
    preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
Else
    preguntas_x_seccion.opB = False
    preguntas_x_seccion.Label3.Caption = ""
End If
If MiTabla!C <> nul Then
    preguntas_x_seccion.opC = True
    preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
Else
    preguntas_x_seccion.opC = False
    preguntas_x_seccion.Label4.Caption = ""
End If
If MiTabla!D <> nul Then
    preguntas_x_seccion.opD = True
    preguntas_x_seccion.Label5.Caption = " " + MiTabla!D
Else
    preguntas_x_seccion.opD = False
    preguntas_x_seccion.Label5.Caption = ""
End If
If MiTabla!E <> nul Then
    preguntas_x_seccion.opE.Visible = True
    preguntas_x_seccion.Label6.Caption = " " + MiTabla!E
Else
    preguntas_x_seccion.opE.Visible = False
    preguntas_x_seccion.Label6.Caption = ""
End If

```

RAMON_A.PRW 4

```
preguntas_x_seccion.opF.Value = True'inicializa "sin contestar"
preguntas_x_seccion.Label7.Caption = Str$(num_aleatorio)
preguntas_x_seccion.Label8.Caption = "sin contestar"
preguntas_x_seccion.Label10.Caption = Str$(Num_pregunta)
Num_pregunta = Num_pregunta + 1
i = i + 1
Tutor_o_Examen.Show
End Sub

Sub Command3D10_Click ()
Materia = "Química"
TipoExamen = -1
Num_pregunta = 1
numero_maximo_de_registros = 228
fin = Timer + 1200 'intervalo del examen en segundos

Set MiBase = OpenDatabase("c:\paquet\dbase\*", False, True, "dbase IV;")
Set MiTabla = MiBase.OpenTable(Materia)
MiTabla.MoveFirst
num_aleatorio = aleatorio(numero_maximo_de_registros)
aleat_anterior = num_aleatorio
For j = 1 To num_aleatorio - 1
    MiTabla.MoveNext
Next j
Respuesta(1).NoPreg = num_aleatorio
Respuesta(1).R = "F" 'sin contestar
Respuesta(1).N = MiTabla!Respuesta
If MiTabla!Pregunta <> nul Then
    preguntas_x_seccion.Label1.Caption = MiTabla!Pregunta
Else
    preguntas_x_seccion.Label1.Caption = ""
End If
If MiTabla!A <> nul Then
    preguntas_x_seccion.opA = True
    preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
Else
    preguntas_x_seccion.opA = False
    preguntas_x_seccion.Label2.Caption = ""
End If
If MiTabla!B <> nul Then
    preguntas_x_seccion.opB = True
    preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
Else
    preguntas_x_seccion.opB = False
    preguntas_x_seccion.Label3.Caption = ""
End If
If MiTabla!C <> nul Then
    preguntas_x_seccion.opC = True
    preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
Else
    preguntas_x_seccion.opC = False
    preguntas_x_seccion.Label4.Caption = ""
End If
If MiTabla!D <> nul Then
    preguntas_x_seccion.opD = True
    preguntas_x_seccion.Label5.Caption = " " + MiTabla!D
Else
```

```

preguntas_x_seccion.opD = False
preguntas_x_seccion.Label5.Caption = ""
End If
If MiTabla!E <> nul Then
preguntas_x_seccion.opE.Visible = True
preguntas_x_seccion.Label6.Caption = " " + MiTabla!E
Else
preguntas_x_seccion.opE.Visible = False
preguntas_x_seccion.Label6.Caption = ""
End If
preguntas_x_seccion.opF.Value = True'inicializa "sin contestar"
preguntas_x_seccion.Label7.Caption = Str$(num_aleatorio)
preguntas_x_seccion.Label8.Caption = "sin contestar"
preguntas_x_seccion.Label10.Caption = Str$(Num_pregunta)
Num_pregunta = Num_pregunta + 1
i = i + 1
'preguntas_x_seccion.Label11.Caption = MiTabla!Pregunta
'preguntas_x_seccion.Label12.Caption = " " + MiTabla!A
'preguntas_x_seccion.Label13.Caption = " " + MiTabla!B
'preguntas_x_seccion.Label14.Caption = " " + MiTabla!C
'preguntas_x_seccion.Label15.Caption = " " + MiTabla!D
'preguntas_x_seccion.Label16.Caption = " " + MiTabla!E
Tutor_o_Examen.Show
End Sub

Sub Command3D10_GotFocus ()
App.HelpFile = "c:\hipertex\quimic.hlp"
End Sub

Sub Command3D11_Click ()
Tipo_Examen.Hide
Examen_x_Seccion.Hide
End Sub

Sub Command3D2_Click ()
Materia = "Español"
TipoExamen = -1
Num_pregunta = 1
numero_maximo_de_registros = 139
fin = Timer + 1200 'intervalo del examen en segundos

Set MiBase = OpenDatabase("c:\paquet\dbase", False, True, "dbase IV;")
Set MiTabla = MiBase.OpenTable(Materia)
MiTabla.MoveFirst
num_aleatorio = aleatorio(numero_maximo_de_registros)
aleat_anterior = num_aleatorio
For j = 1 To num_aleatorio - 1
MiTabla.MoveNext
Next j
Respuesta(1).NoPreg = num_aleatorio
Respuesta(1).R = "F" 'sin contestar
Respuesta(1).N = MiTabla!Respuesta
If MiTabla!Pregunta <> nul Then
preguntas_x_seccion.Label11.Caption = MiTabla!Pregunta
Else
preguntas_x_seccion.Label11.Caption = ""
End If

```

EXAMEN_A.FRM - *

```

If MiTabla!A <> nul Then
    preguntas_x_seccion.opA = True
    preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
Else
    preguntas_x_seccion.opA = False
    preguntas_x_seccion.Label2.Caption = ""
End If
If MiTabla!B <> nul Then
    preguntas_x_seccion.opB = True
    preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
Else
    preguntas_x_seccion.opB = False
    preguntas_x_seccion.Label3.Caption = ""
End If
If MiTabla!C <> nul Then
    preguntas_x_seccion.opC = True
    preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
Else
    preguntas_x_seccion.opC = False
    preguntas_x_seccion.Label4.Caption = ""
End If
If MiTabla!D <> nul Then
    preguntas_x_seccion.opD = True
    preguntas_x_seccion.Label5.Caption = " " + MiTabla!D
Else
    preguntas_x_seccion.opD = False
    preguntas_x_seccion.Label5.Caption = ""
End If
If MiTabla!E <> nul Then
    preguntas_x_seccion.opE.Visible = True
    preguntas_x_seccion.Label6.Caption = " " + MiTabla!E
Else
    preguntas_x_seccion.opE.Visible = False
    preguntas_x_seccion.Label6.Caption = ""
End If
preguntas_x_seccion.opF.Value = True'inicializa "sin contestar"
preguntas_x_seccion.Label7.Caption = Str$(num_aleatorio)
preguntas_x_seccion.Label8.Caption = "sin contestar"
preguntas_x_seccion.Label10.Caption = Str$(Num_pregunta)
Num_pregunta = Num_pregunta + 1
i = i + 1
'preguntas_x_seccion.Label11.Caption = MiTabla!Pregunta
'preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
'preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
'preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
'preguntas_x_seccion.Label5.Caption = " " + MiTabla!D
'preguntas_x_seccion.Label6.Caption = " " + MiTabla!E
'preguntas_gauge.Show
Tutor_o_Examen.Show
End Sub

Sub Command3D2_GotFocus ()
    App.HelpFile = "c:\hipertex\cwh.hlp"
End Sub

Sub Command3D3_Click ()
    Materia = "Fisica"
    TipoExamen = -1

```



```

Set MiBase = OpenDatabase("c:\paquet\dbase", False, True, "dbase IV;")
Set MiTabla = MiBase.OpenTable(Materia)
MiTabla.MoveFirst
preguntas_x_seccion.Label1.Caption = MiTabla!Pregunta
preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
preguntas_x_seccion.Label5.Caption = " " + MiTabla!D
'Label6.Caption = MiTabla!E
preguntas_x_seccion.Show 1

```

End Sub

```

Sub Command3D4_Click ()
Materia = "Geo_gral"
TipoExamen = -1
numero_maximo_de_registros = 388
Num_pregunta = 1
fin = Timer + 1200 'intervalo del examen en segundos
Set MiBase = OpenDatabase("c:\paquet\dbase", False, True, "dbase IV;")
Set MiTabla = MiBase.OpenTable(Materia)
MiTabla.MoveFirst
num_aleatorio = aleatorio(numero_maximo_de_registros)
aleat_anterior = num_aleatorio
For j = 1 To num_aleatorio - 1
    MiTabla.MoveNext
Next j
Respuesta(1).NoPreg = num_aleatorio
Respuesta(1).R = "P" sin Contestar
Respuesta(1).N = MiTabla!Respuesta
If MiTabla!Pregunta <> nul Then
    preguntas_x_seccion.Label1.Caption = MiTabla!Pregunta
Else
    preguntas_x_seccion.Label1.Caption = ""
End If
If MiTabla!A <> nul Then
    preguntas_x_seccion.opA = True
    preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
Else
    preguntas_x_seccion.opA = False
    preguntas_x_seccion.Label2.Caption = ""
End If
If MiTabla!B <> nul Then
    preguntas_x_seccion.opB = True
    preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
Else
    preguntas_x_seccion.opB = False
    preguntas_x_seccion.Label3.Caption = ""
End If
If MiTabla!C <> nul Then
    preguntas_x_seccion.opC = True
    preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
Else
    preguntas_x_seccion.opC = False
    preguntas_x_seccion.Label4.Caption = ""
End If
If MiTabla!D <> nul Then
    preguntas_x_seccion.opD = True
    preguntas_x_seccion.Label5.Caption = " " + MiTabla!D

```

EXAMEN_3.PRW * *

```
Else
    preguntas_x_seccion.opD = False
    preguntas_x_seccion.Label5.Caption = ""
End If
If MiTabla!E <> nul Then
    preguntas_x_seccion.opE.Visible = True
    preguntas_x_seccion.Label6.Caption = " " + MiTabla!E
Else
    preguntas_x_seccion.opE.Visible = False
    preguntas_x_seccion.Label6.Caption = ""
End If
preguntas_x_seccion.opF.Value = True'inicializa "sin contestar"
preguntas_x_seccion.Label7.Caption = Str$(num_aleatorio)
preguntas_x_seccion.Label8.Caption = "sin contestar"
preguntas_x_seccion.Label10.Caption = Str$(Num_pregunta)
Num_pregunta = Num_pregunta + 1
i = i + 1
Tutor_o_Examen.Show
End Sub

Sub Command3D5_Click ()
Materia = "Geo_mex"
TipoExamen = -1
numero_maximo_de_registros = 143
Num_pregunta = 1
fin = Timer + 1200 'intervalo del examen en segundos
Set MiBase = OpenDatabase("c:\paquet\dbase\*", False, True, "dbase IV;")
Set MiTabla = MiBase.OpenTable(Materia)
MiTabla.MoveFirst
num_aleatorio = aleatorio(numero_maximo_de_registros)
aleat_anterior = num_aleatorio
For j = 1 To num_aleatorio - 1
    MiTabla.MoveNext
Next j
Respuesta(1).NoPreg = num_aleatorio
Respuesta(1).R = "F" 'sin contestar
Respuesta(1).H = MiTabla!Respuesta
If MiTabla!Pregunta <> nul Then
    preguntas_x_seccion.Label11.Caption = MiTabla!Pregunta
Else
    preguntas_x_seccion.Label11.Caption = ""
End If
If MiTabla!A <> nul Then
    preguntas_x_seccion.opA = True
    preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
Else
    preguntas_x_seccion.opA = False
    preguntas_x_seccion.Label2.Caption = ""
End If
If MiTabla!B <> nul Then
    preguntas_x_seccion.opB = True
    preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
Else
    preguntas_x_seccion.opB = False
    preguntas_x_seccion.Label3.Caption = ""
End If
If MiTabla!C <> nul Then
    preguntas_x_seccion.opC = True
    preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
```

```

Else
    preguntas_x_seccion.opC = False
    preguntas_x_seccion.Label4.Caption = ""
End If
If MiTabla!D <> nul Then
    preguntas_x_seccion.opD = True
    preguntas_x_seccion.Label5.Caption = " " + MiTabla!D
Else
    preguntas_x_seccion.opD = False
    preguntas_x_seccion.Label5.Caption = ""
End If
If MiTabla!E <> nul Then
    preguntas_x_seccion.opE.Visible = True
    preguntas_x_seccion.Label6.Caption = " " + MiTabla!E
Else
    preguntas_x_seccion.opE.Visible = False
    preguntas_x_seccion.Label6.Caption = ""
End If
preguntas_x_seccion.opF.Value = True 'inicializa "sin contestar"
preguntas_x_seccion.Label7.Caption = Str$(num_aleatorio)
preguntas_x_seccion.Label8.Caption = "sin contestar"
preguntas_x_seccion.Label10.Caption = Str$(Num_preguntas)
Num_pregunta = Num_pregunta + 1
i = i + 1
Tutor_o_Examen.Show
End Sub

Sub Command306_Click ()
Materia = "Hist_Mex"
TipoExamen = -1
Num_pregunta = 1
numero_maximo_de_registros = 194
fin = Timer + 1200 'intervalo del examen en segundos
Set MiBase = OpenDatabase("c:\paquet\ibase\*", False, True, "ibase IV,")
Set MiTabla = MiBase.OpenTable(Materia)
MiTabla.MoveFirst
num_aleatorio = aleatorio(numero_maximo_de_registros)
aleat_anterior = num_aleatorio
For j = 1 To num_aleatorio - 1
    MiTabla.MoveNext
Next j
Respuesta(1).NoPreg = num_aleatorio
Respuesta(1).R = "F" 'sin contestar
Respuesta(1).N = MiTabla!Respuesta
If MiTabla!Pregunta <> nul Then
    preguntas_x_seccion.Label11.Caption = MiTabla!Pregunta
Else
    preguntas_x_seccion.Label11.Caption = ""
End If
If MiTabla!A <> nul Then
    preguntas_x_seccion.opA = True
    preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
Else
    preguntas_x_seccion.opA = False
    preguntas_x_seccion.Label2.Caption = ""
End If
If MiTabla!B <> nul Then
    preguntas_x_seccion.opB = True

```

*****A.FUN - 0

```
preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
Else
preguntas_x_seccion.opB = False
preguntas_x_seccion.Label3.Caption = ""
End If
If MiTabla!C <> nul Then
preguntas_x_seccion.opC = True
preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
Else
preguntas_x_seccion.opC = False
preguntas_x_seccion.Label4.Caption = ""
End If
If MiTabla!D <> nul Then
preguntas_x_seccion.opD = True
preguntas_x_seccion.Label5.Caption = " " + MiTabla!D
Else
preguntas_x_seccion.opD = False
preguntas_x_seccion.Label5.Caption = ""
End If
If MiTabla!E <> nul Then
preguntas_x_seccion.opE.Visible = True
preguntas_x_seccion.Label6.Caption = " " + MiTabla!E
Else
preguntas_x_seccion.opE.Visible = False
preguntas_x_seccion.Label6.Caption = ""
End If
preguntas_x_seccion.opF.Value = True 'inicializa "sin contestar"
preguntas_x_seccion.Label7.Caption = Str$(num_aleatorio)
preguntas_x_seccion.Label8.Caption = "sin contestar"
preguntas_x_seccion.Label10.Caption = Str$(Num_pregunta)
Num_pregunta = Num_pregunta + 1
i = i + 1
Tutor_o_Examen.Show
End Sub

Sub Command3D7_Click ()
Materia = "Hist_Uni"
TipoExamen = -1

Set MiBase = OpenDatabase("c:\paquet\dbase\"," , False, True, "dbase IV,")
Set MiTabla = MiBase.OpenTable(Materia)
MiTabla.MoveFirst

preguntas_x_seccion.Label1.Caption = MiTabla!Pregunta
preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
preguntas_x_seccion.Label5.Caption = " " + MiTabla!D
'Label6.Caption = MiTabla!E
preguntas_x_seccion.Show 1
End Sub

Sub Command3D8_Click ()
Materia = "Literatu"
TipoExamen = -1
Num_pregunta = 1
numero_maximo_de_registros = 107
fin = Timer + 1200 'intervalo del examen en segundos
```

```

Set MiBase = OpenDatabase("c:\paquet\ibase\", False, True, "dbase IV;")
Set MiTabla = MiBase.OpenTable(Materia)
MiTabla.MoveFirst
num_aleatorio = aleatorio(numero_maximo_de_registros)
aleat_anterior = num_aleatorio
For j = 1 To num_aleatorio - 1
    MiTabla.MoveNext
Next j
Respuesta(1).NoPreg = num_aleatorio
Respuesta(1).R = "F" sin contestar
Respuesta(1).N = MiTabla!Respuesta
If MiTabla!Preguntas <> nul Then
    preguntas_x_seccion.Label1.Caption = MiTabla!Preguntas
Else
    preguntas_x_seccion.Label1.Caption = ""
End If
If MiTabla!A <> nul Then
    preguntas_x_seccion.opA = True
    preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
Else
    preguntas_x_seccion.opA = False
    preguntas_x_seccion.Label2.Caption = ""
End If
If MiTabla!B <> nul Then
    preguntas_x_seccion.opB = True
    preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
Else
    preguntas_x_seccion.opB = False
    preguntas_x_seccion.Label3.Caption = ""
End If
If MiTabla!C <> nul Then
    preguntas_x_seccion.opC = True
    preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
Else
    preguntas_x_seccion.opC = False
    preguntas_x_seccion.Label4.Caption = ""
End If
If MiTabla!D <> nul Then
    preguntas_x_seccion.opD = True
    preguntas_x_seccion.Label5.Caption = " " + MiTabla!D
Else
    preguntas_x_seccion.opD = False
    preguntas_x_seccion.Label5.Caption = ""
End If
If MiTabla!E <> nul Then
    preguntas_x_seccion.opE.Visible = True
    preguntas_x_seccion.Label6.Caption = " " + MiTabla!E
Else
    preguntas_x_seccion.opE.Visible = False
    preguntas_x_seccion.Label6.Caption = ""
End If
preguntas_x_seccion.opF.Value = True'inicializa "sin contestar"
preguntas_x_seccion.Label7.Caption = Str$(num_aleatorio)
preguntas_x_seccion.Label8.Caption = "sin contestar"
preguntas_x_seccion.Label10.Caption = Str$(Num_preguntas)
Num_pregunta = Num_pregunta + 1
i = i + 1

```

MANEJAR_A.PRG - 10

```
'preguntas_x_seccion.Label1.Caption = MiTabla!Pregunta
'preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
'preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
'preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
'preguntas_x_seccion.Label5.Caption = " " + MiTabla!D
'preguntas_x_seccion.Label6.Caption = " " + MiTabla!E
'preguntas_gauge.Show
Tutor_o_Examen.Show
End Sub

Sub Command3D8_GotFocus ()
  App.HelpFile = "c:\hipertex\literat.hlp"
End Sub

Sub Command3D9_Click ()
  Materia = "Matemati"
  TipoExamen = -1

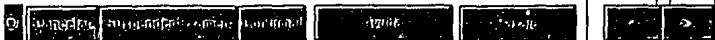
  Set MiBase = OpenDatabase("c:\paquet\dbase\*", False, True, "dbase IV;")
  Set MiTabla = MiBase.OpenTable(Materia)
  MiTabla.MoveFirst

  preguntas_x_seccion.Label1.Caption = MiTabla!Pregunta
  preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
  preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
  preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
  preguntas_x_seccion.Label5.Caption = " " + MiTabla!D
  'Label6.Caption = MiTabla!E
  preguntas_x_seccion.Show 1
End Sub
```



List4

List1	13



PRECSECC.FRM - 1

```
Dim Pregunta_bien As String
Dim texto_preg(1 To 256) As String * 1
Dim imin As Integer
Dim str_aux As String
```

```
Sub AniButton4_Click ()
intermitente =(intermitente) * (-1)
If intermitente = -1 Then
    command3d1.Visible = False
    command3d2.Visible = False
    list1.Visible = True
    list2.Visible = True
    list3.Visible = True
    command1.Visible = True
    command2.Visible = True
Else
    command3d1.Visible = True
    command3d2.Visible = True
    list1.Visible = False
    list2.Visible = False
    list3.Visible = False
    command1.Visible = False
    command2.Visible = False

```

```
End If
End Sub
```

```
Sub Cancelar_Click ()
confirmar.Enabled = False
cancelar.Enabled = False
End Sub
```

```
Sub Command1_Click ()
SendKeys "{up}"
num = list3.ListCount
If list3.ListIndex > 0 Then
    list3.ListIndex = list3.ListIndex - 1
End If
list2.ListIndex = list3.ListIndex
list1.ListIndex = list3.ListIndex
Select Case list2
    Case "*"
        Case "a" opF.Value = True
        Case "a" opA.Value = True
        Case "b"
            opB.Value = True
        Case "c"
            opC.Value = True
        Case "d"
            opD.Value = True
        Case "e"
            opE.Value = True
End Select
End Sub
```

```
Sub Command2_Click ()
'SendKeys "{down}"
list3.SetFocus
```


PRECSECC.FRM - 2

```
num = list3.ListCount
If list3.ListIndex < num - 1 Then
    list3.ListIndex = list3.ListIndex + 1
End If
list2.ListIndex = list3.ListIndex
list1.ListIndex = list3.ListIndex
```

```
Select Case list2
    Case "*"
        opF.Value = True
    Case "a"
        opA.Value = True
    Case "b"
        opB.Value = True
    Case "c"
        opC.Value = True
    Case "d"
        opD.Value = True
    Case "e"
        opE.Value = True
End Select
```

End Sub

```
Sub Command3D1_Click ()
If list3.ListIndex < 219 Then
    list3.ListIndex = list3.ListIndex + 1
End If
End Sub
```

```
Sub Command3D2_Click ()
If list3.ListIndex >= 1 Then
list3.ListIndex = list3.ListIndex - 1
End If
End Sub
```

```
Sub Command3D3_Click ()
confirmar.Enabled = True
cancelar.Enabled = True
End Sub
```

```
Sub Confirmar_Click ()
salvar_suspendido.Show
End Sub
```

```
Sub Form_Load ()
    TipoExamen = -1
    contador = 1
    intermitente = -1
    command3d1.Visible = True
    command3d2.Visible = True
    list1.Visible = False
    list2.Visible = False
    list3.Visible = False
    command1.Visible = False
    command2.Visible = False
    pasaje.Enabled = False
    list1.Enabled = False
    list2.Enabled = False
    aux = contador
    Picture1.Picture = LoadPicture()
```

PRECSECC.FRM - 3

```
Picture1.Visible = False
For i1 = aux To aux + 29
  Do
    repetido = 0
    preg_aleatorio = aleatorio(numero_maximo_de_registros)
    For i2 = aux To aux + 29
      If preg_aleatorio = arreglo_global(i2).num_preg_global Then
        repetido = -1
        Exit For i2
      Else
        If arreglo_global(i1).num_preg_global = 0 Then
          arreglo_global(i1).num_preg_global = preg_aleatorio
          arreglo_global(i1).Materia_global = Materia
          arreglo_global(i1).resp_global = "*"
          arreglo_global(i1).contador_global = contador
          list5.AddItem (Str(arreglo_global(i1).num_preg_global))
          list3.AddItem (Str(arreglo_global(i1).contador_global))
          list2.AddItem arreglo_global(i1).resp_global
          list1.AddItem arreglo_global(i1).Materia_global
          contador = contador + 1
          Exit For i2
        Else
          Exit For i2
        End If
      End If
    Next i2
  Loop Until repetido = 0
Next i1
End Sub
```

```
Sub List3_Click ()
  Set MiBase = OpenDatabase("c:\paquet\dbase\", False, True, "dbase IV;")
  Materia = LTrim(RTrim(arreglo_global(list3.ListIndex + 1).Materia_global))
)
  Set MiTabla = MiBase.OpenTable(Materia)
  MiTabla.Index = LTrim(RTrim(Materia))
  num_aleatorio = arreglo_global(list3.ListIndex + 1).num_preg_global
  Select Case Materia
    Case "Espanol"
      MiTabla.Seek "=", num_aleatorio + 1000000
    Case "Literatu"
      MiTabla.Seek "=", num_aleatorio + 2000000
    Case "Hist Uni"
      MiTabla.Seek "=", num_aleatorio + 3000000
    Case "Hist mex"
      MiTabla.Seek "=", num_aleatorio + 4000000
    Case "Geo gral"
      MiTabla.Seek "=", num_aleatorio + 5000000
    Case "Geo mex"
      MiTabla.Seek "=", num_aleatorio + 6000000
    Case "Math"
      MiTabla.Seek "=", num_aleatorio + 7000000
    Case "Fisic"
      MiTabla.Seek "=", num_aleatorio + 8000000
    Case "Quimica"
      MiTabla.Seek "=", num_aleatorio + 9000000
    Case "Biologia"
      MiTabla.Seek "=", num_aleatorio + 0
  End Select
  Respuesta(list3.ListIndex + 1).N = MiTabla!Respuesta
  If MiTabla!Formula = 0 Then
```

```

Label1.Caption = ""
Label2.Caption = ""
Label3.Caption = ""
Label4.Caption = ""
Label5.Caption = ""
Label6.Caption = ""
Label1.Visible = True
Label2.Visible = True
Label3.Visible = True
Label4.Visible = True
Label5.Visible = True
Label6.Visible = True
Picture1.Picture = LoadPicture()
Picture1.Visible = False

If MiTabla!Pregunta <> nul Then
    imin = 1
    str_aux = MiTabla!Pregunta
    For imid = 1 To 256
        texto_preg(imid) = Mid$(str_aux, imid, 1)
        If texto_preg(imid) = "!" OR texto_preg(imid) = "1" Then
            Pregunta_bien = Pregunta_bien + Chr(10)
        Else
            Pregunta_bien = Pregunta_bien + texto_preg(imid)
        End If
    Next imid
    Label1.Caption = Pregunta_bien
    Pregunta_bien = ""
Else
    Label1.Caption = ""
End If
If MiTabla!A <> nul Then
    opA.Visible = True
    Label2.Caption = " " + MiTabla!A
Else
    Label2.Caption = ""
    opA.Visible = False
End If
If MiTabla!B <> nul Then
    opB.Visible = True
    Label3.Caption = " " + MiTabla!B
Else
    Label3.Caption = ""
    opB.Visible = False
End If
If MiTabla!C <> nul Then
    opC.Visible = True
    Label4.Caption = " " + MiTabla!C
Else
    Label4.Caption = ""
    opC.Visible = False
End If
If MiTabla!D <> nul Then
    opD.Visible = True
    Label5.Caption = " " + MiTabla!D
Else
    Label5.Caption = ""
    opD.Visible = False
End If
If MiTabla!E <> nul Then

```

```

        opE.Visible = True
        Label6.Caption = " " + MiTabla!E
    Else
        Label6.Caption = ""
        opE.Visible = False
    End If
    If MiTabla!N_PASAJE > 0 Then
        pasaje.Enabled = True
        lentes.Enabled = True
        pasaje.HelpContextID = MiTabla!N_PASAJE
        lentes.HelpContextID = MiTabla!N_PASAJE
        App.HelpFile = "c:\hipertex\pasaje.hlp"
    Else
        pasaje.Enabled = False
        lentes.Enabled = True
    End If
    opF.Value = True
    list2.ListIndex = list3.ListIndex
    Select Case list2
        Case "*"
            opF.Value = True
        Case "a"
            opA.Value = True
        Case "b"
            opB.Value = True
        Case "c"
            opC.Value = True
        Case "d"
            opD.Value = True
        Case "e"
            opE.Value = True
    End Select
Else
    Select Case Materia
        Case "Math"
            MiTabla.Seek "=", num_aleatorio + 7000000
            pathbase = "c:\mate\mEf\"
            extnfile = ".WMF"
        Case "Fisic"
            MiTabla.Seek "=", num_aleatorio + 8000000
            pathbase = "c:\fisi\mEf\"
            extnfile = ".WMF"
        Case "Quimica"
            MiTabla.Seek "=", num_aleatorio + 9000000
            pathbase = "c:\powerpnt\"
            extnfile = ".WMF"
    End Select
    Label1.Caption = ""
    Label2.Caption = ""
    Label3.Caption = ""
    Label4.Caption = ""
    Label5.Caption = ""
    Label6.Caption = ""
    Label1.Visible = False
    Label2.Visible = False
    Label3.Visible = False
    Label4.Visible = False
    Label5.Visible = False
    Label6.Visible = False
    Picture1.Picture = LoadPicture()
    Picture1.Visible = True

```

```

If MiTabla1ArchiPre <> nul Then
  str_aux = LTrim(RTrim(MiTabla1ArchiPre))
  pathbase = LTrim(RTrim(pathbase))
  extnfile = LTrim(RTrim(extnfile))
  Picture1.Picture = LoadPicture(pathbase + str_aux + extnfile)
Else
  Picture1.Picture = LoadPicture()
End If
End Sub

Sub List3_DblClick ()
  Set MiBase = OpenDatabase("c:\paquet\dbase\", False, True, "dbase IV;")
  Materia = LTrim(RTrim(arreglo_global(list3.ListIndex + 1).Materia_globa
))

  Set MiTabla = MiBase.OpenTable(Materia)
  MiTabla.Index = LTrim(RTrim((Materia)))
  num_aleatorio = arreglo_global(list3.ListIndex + 1).num_preg_global
  Select Case Materia
    Case "Espanol"
      MiTabla.Seek "=", num_aleatorio + 1000000
    Case "Literatu"
      MiTabla.Seek "=", num_aleatorio + 2000000
    Case "Hist Uni"
      MiTabla.Seek "=", num_aleatorio + 3000000
    Case "Hist mex"
      MiTabla.Seek "=", num_aleatorio + 4000000
    Case "Geo gral"
      MiTabla.Seek "=", num_aleatorio + 5000000
    Case "Geo mex"
      MiTabla.Seek "=", num_aleatorio + 6000000
    Case "Math"
      MiTabla.Seek "=", num_aleatorio + 7000000
    Case "Fisic"
      MiTabla.Seek "=", num_aleatorio + 8000000
    Case "Quimica"
      MiTabla.Seek "=", num_aleatorio + 9000000
    Case "Biologia"
      MiTabla.Seek "=", num_aleatorio + 0
  End Select
  Respuesta(i).N = MiTabla1Respuesta
If MiTabla1Formula = 0 Then
  Label1.Caption = ""
  Label2.Caption = ""
  Label3.Caption = ""
  Label4.Caption = ""
  Label5.Caption = ""
  Label6.Caption = ""
  Label1.Visible = True
  Label2.Visible = True
  Label3.Visible = True
  Label4.Visible = True
  Label5.Visible = True
  Label6.Visible = True
  Picture1.Picture = LoadPicture()
  Picture1.Visible = False

```

```

If MiTabla!Pregunta <> nul Then
    imin = 1
    str_aux = MiTabla!Pregunta
    For imid = 1 To 256
        texto_preg(imid) = Mid$(str_aux, imid, 1)
        If texto_preg(imid) = "r" Or texto_preg(imid) = "l" Then
            Pregunta_bien = Pregunta_bien + Chr(10)
        Else
            Pregunta_bien = Pregunta_bien + texto_preg(imid)
        End If
    Next imid
    Label1.Caption = Pregunta_bien
    Pregunta_bien = ""
Else
    Label1.Caption = ""
End If
If MiTabla!A <> nul Then
    opA.Visible = True
    Label2.Caption = " " + MiTabla!A
Else
    Label2.Caption = ""
    opA.Visible = False
End If
If MiTabla!B <> nul Then
    opB.Visible = True
    Label3.Caption = " " + MiTabla!B
Else
    Label3.Caption = ""
    opB.Visible = False
End If
If MiTabla!C <> nul Then
    opC.Visible = True
    Label4.Caption = " " + MiTabla!C
Else
    Label4.Caption = ""
    opC.Visible = False
End If
If MiTabla!D <> nul Then
    opD.Visible = True
    Label5.Caption = " " + MiTabla!D
Else
    Label5.Caption = ""
    opD.Visible = False
End If
If MiTabla!E <> nul Then
    opE.Visible = True
    Label6.Caption = " " + MiTabla!E
Else
    Label6.Caption = ""
    opE.Visible = False
End If
If MiTabla!N PASAJE > 0 Then
    pasaje.Enabled = True
    lentes.Enabled = True
    pasaje.HelpContextID = MiTabla!N PASAJE
    lentes.HelpContextID = MiTabla!N PASAJE
    App.HelpFile = "c:\hipertex\pasaje.hlp"
Else
    pasaje.Enabled = False

```

```

        lentos.Enabled = True
    End If
    Select Case list2
    Case "*"
        opF.Value = True
    Case "a"
        opA.Value = True
    Case "b"
        opB.Value = True
    Case "c"
        opC.Value = True
    Case "d"
        opD.Value = True
    Case "e"
        opE.Value = True
    End Select
Else
    Select Case Materia
    Case "Math"
        MiTabla.Seek "=", num_aleatorio + 7000000
        pathbase = "c:\mate\mEf\"
        extnfile = ".WMF"
    Case "Fisic"
        MiTabla.Seek "=", num_aleatorio + 8000000
        pathbase = "c:\fisi\mEf\"
        extnfile = ".WMF"
    Case "Quimica"
        MiTabla.Seek "=", num_aleatorio + 9000000
        pathbase = "c:\powerpnt\"
        extnfile = ".WMF"
    End Select
    Label1.Caption = ""
    Label2.Caption = ""
    Label3.Caption = ""
    Label4.Caption = ""
    Label5.Caption = ""
    Label6.Caption = ""
    Label1.Visible = False
    Label2.Visible = False
    Label3.Visible = False
    Label4.Visible = False
    Label5.Visible = False
    Label6.Visible = False
    Picture1.Picture = LoadPicture()
    Picture1.Visible = True
    If MiTabla!ArchiPre <> nul Then
        str_aux = LTrim(RTrim(MiTabla!ArchiPre))
        pathbase = LTrim(RTrim(pathbase))
        extnfile = LTrim(RTrim(extnfile))
        Picture1.Picture = LoadPicture(pathbase + str_aux + extnfile)
    Else
        Picture1.Picture = LoadPicture()
    End If
    opF.Value = True'inicializa sin contestar
    'label10.Caption = Str$(Num_pregunta)
    Num_pregunta = Num_pregunta + 1
    aleat_anterior = num_aleatorio
End If
End Sub

Sub List3_KeyDown (KeyCode As Integer, Shift As Integer)

```

PRECSECC.FRM - 9

```
list2.ListIndex = list3.ListIndex
list1.ListIndex = list3.ListIndex
list5.ListIndex = list3.ListIndex
End Sub

Sub List3_KeyUp (KeyCode As Integer, Shift As Integer)
list2.ListIndex = list3.ListIndex
list1.ListIndex = list3.ListIndex
list5.ListIndex = list3.ListIndex
End Sub

Sub List3_MouseMove (Button As Integer, Shift As Integer, X As Single, Y As Single)
list2.ListIndex = list3.ListIndex
list1.ListIndex = list3.ListIndex
list5.ListIndex = list3.ListIndex
End Sub

Sub opA_Click (Value As Integer)
list2.Enabled = True
list2.RemoveItem list3.ListIndex
list2.AddItem "a", list3.ListIndex
list2.ListIndex = list3.ListIndex
list2.Enabled = False
End Sub

Sub opB_Click (Value As Integer)
list2.Enabled = True
list2.RemoveItem list3.ListIndex
list2.AddItem "b", list3.ListIndex
list2.ListIndex = list3.ListIndex
list2.Enabled = False
End Sub

Sub opC_Click (Value As Integer)
list2.Enabled = True
list2.RemoveItem list3.ListIndex
list2.AddItem "c", list3.ListIndex
list2.ListIndex = list3.ListIndex
list2.Enabled = False
End Sub

Sub opD_Click (Value As Integer)
list2.Enabled = True
list2.RemoveItem list3.ListIndex
list2.AddItem "d", list3.ListIndex
list2.ListIndex = list3.ListIndex
list2.Enabled = False
End Sub

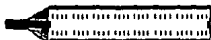
Sub opE_Click (Value As Integer)
list2.Enabled = True
list2.RemoveItem list3.ListIndex
list2.AddItem "e", list3.ListIndex
list2.ListIndex = list3.ListIndex
list2.Enabled = False
```


PRECSECC.FRM - 10

End Sub

Label1 Label7

Label8



de 30



```

Sub Command3D4_Click ()
App.HelpFile = "c:\hipertex\LITERATURA.hlp"
' AppActivate "ayuda"
SendKeys "{f1}"
End Sub

```

```

Sub Cancelar_Click ()
confirmar.Enabled = False
cancelar.Enabled = False
End Sub

```

```

Sub Command3D1_Click ()
    pasaje.Enabled = False
    MiTabla.Index = LTrim(RTrim(Materia))
    If Num_pregunta < 31 Then
    Do
        ya_esta_en_lista = 0
        num_aleatorio = aleatorio(numero_maximo_de_registros)
        For i = 2 To Num_pregunta
            If num_aleatorio = Respuesta(i).NoPreg Then
                ya_esta_en_lista = -1
                Exit For
            Else
                If Respuesta(i).NoPreg = 0 Then
                    Respuesta(i).NoPreg = num_aleatorio
                    Exit For
                End If
            End If
        Next i
    Loop Until ya_esta_en_lista = 0
    Else
        salvar.Show
    End If
    preguntas_x_seccion.Label7.Caption = Str$(num_aleatorio)
    Select Case Materia
        Case "Español"
            MiTabla.Seek "=", num_aleatorio + 1000000
        Case "Literatu"
            MiTabla.Seek "=", num_aleatorio + 2000000
        Case "hissuni"
            MiTabla.Seek "=", num_aleatorio + 3000000
        Case "Hist_mex"
            MiTabla.Seek "=", num_aleatorio + 4000000
        Case "Geo_gral"
            MiTabla.Seek "=", num_aleatorio + 5000000
        Case "Geo_mex"
            MiTabla.Seek "=", num_aleatorio + 6000000
        Case "Haaaattee"
            MiTabla.Seek "=", num_aleatorio + 7000000
        Case "fiiisica"
            MiTabla.Seek "=", num_aleatorio + 8000000
        Case "Química"
            MiTabla.Seek "=", num_aleatorio + 9000000
        Case "Biología"
    
```

```

        MiTabla.Seek "=", num_aleatorio + 0

End Select
Respuesta(i).N = MiTabla!Respuesta
If MiTabla!Pregunta <> nul Then
    Label1.Caption = MiTabla!Pregunta
Else
    Label1.Caption = ""
End If

If MiTabla!A <> nul Then
    opA.Visible = True
    Label2.Caption = " " + MiTabla!A
Else
    Label2.Caption = ""
    opA.Visible = False
End If

If MiTabla!B <> nul Then
    opB.Visible = True
    Label3.Caption = " " + MiTabla!B
Else
    Label3.Caption = ""
    opB.Visible = False
End If

If MiTabla!C <> nul Then
    opC.Visible = True
    Label4.Caption = " " + MiTabla!C
Else
    Label4.Caption = ""
    opC.Visible = False
End If

If MiTabla!D <> nul Then
    opD.Visible = True
    Label5.Caption = " " + MiTabla!D
Else
    Label5.Caption = ""
    opD.Visible = False
End If

If MiTabla!E <> nul Then
    opE.Visible = True
    Label6.Caption = " " + MiTabla!E
Else
    Label6.Caption = ""
    opE.Visible = False
End If

'If MiTabla!N_PASAJE > 0 Then
'    pasaje.Enabled = True
'    pasaje.HelpContextID = MiTabla!N_PASAJE
'    App.HelpFile = "c:\hipertex\pasaje.hlp"
'End If

MiTabla.Index = LTrim(RTrim(Materia))
Select Case Materia
Case "Espanol"
    MiTabla.Seek "=", aleat_anterior + 1000000
Case "Literatu"
    MiTabla.Seek "=", aleat_anterior + 2000000
'Case "hisssuini"
'    MiTabla.Seek "=", aleat_anterior + 3000000
'Case "hisssmex"
'    MiTabla.Seek "=", aleat_anterior + 4000000

```

```

Case "Geo_gral"
    MiTabla.Seek "=", aleat_anterior + 500000
Case "Geo_mex"
    MiTabla.Seek "=", aleat_anterior + 6000000
'Case "Maaantes"
    ' MiTabla.Seek "=", aleat_anterior + 7000000
'Case "fiisica"
    ' MiTabla.Seek "=", aleat_anterior + 8000000
Case "Quimica"
    MiTabla.Seek "=", aleat_anterior + 9000000
End Select
If (opA.Value = True) And (MiTabla!Respuesta = "a") Then
    gauge1.Value = gauge1.Value + 1
End If
If (opB.Value = True) And (MiTabla!Respuesta = "b") Then
    gauge1.Value = gauge1.Value + 1
End If
If (opC.Value = True) And (MiTabla!Respuesta = "c") Then
    gauge1.Value = gauge1.Value + 1
End If
If (opD.Value = True) And (MiTabla!Respuesta = "d") Then
    gauge1.Value = gauge1.Value + 1
End If
If (opE.Value = True) And (MiTabla!Respuesta = "e") Then
    gauge1.Value = gauge1.Value + 1
End If
opF.Value = True'inicializa sin contestar
label8.Caption = "Sin contestar"
label10.Caption = Str$(Num_pregunta)
Num_pregunta = Num_pregunta + 1
aleat_anterior = num_aleatorio
End Sub

Sub Command3D3_Click ()
    confirmar.Enabled = True
    cancelar.Enabled = True
End Sub

Sub Confirmar_Click ()
    Suspendor_Tutor.Show
End Sub

Sub Form_Load ()
    buense = 0
    pasaje.Enabled = False
    i = 1
    NumPregunta = 1
    ayuda.Enabled = True
If TipoExamen = 0 Then 'global
    Set MiBase = OpenDatabase("c:\paquet\dbase\*", False, True, "dbase IV;")
    Set MiTabla = MiBase.OpenTable(LTrim(RTrim(Materia)))
    MiTabla.MoveFirst
    'label11.Caption = MiTabla!Pregunta
    'label12.Caption = " " + MiTabla!A
    'label13.Caption = " " + MiTabla!B
    'label14.Caption = " " + MiTabla!C
    'label15.Caption = " " + MiTabla!D

```

```

If MiTabla!Pregunta <> nul Then
    preguntas_x_seccion.Label1.Caption = MiTabla!Pregunta
Else
    preguntas_x_seccion.Label1.Caption = ""
End If
If MiTabla!A <> nul Then
    preguntas_x_seccion.opA = True
    preguntas_x_seccion.Label2.Caption = " " + MiTabla!A
Else
    preguntas_x_seccion.opA = False
    Label2.Caption = ""
End If
If MiTabla!B <> nul Then
    preguntas_x_seccion.opB = True
    preguntas_x_seccion.Label3.Caption = " " + MiTabla!B
Else
    preguntas_x_seccion.opB = False
    Label3.Caption = ""
End If
If MiTabla!C <> nul Then
    preguntas_x_seccion.opC = True
    preguntas_x_seccion.Label4.Caption = " " + MiTabla!C
Else
    preguntas_x_seccion.opC = False
    Label4.Caption = ""
End If
If MiTabla!D <> nul Then
    preguntas_x_seccion.opD = True
    preguntas_x_seccion.Label5.Caption = " " + MiTabla!D
Else
    preguntas_x_seccion.opD = False
    Label5.Caption = ""
End If
If MiTabla!E <> nul Then
    preguntas_x_seccion.opE = True
    preguntas_x_seccion.Label6.Caption = " " + MiTabla!E
Else
    preguntas_x_seccion.opE = False
    Label6.Caption = ""
End If
'If MiTabla!N_PASAJE > 0 Then
'    pasaje.Enabled = True
'    pasaje.HelpContextID = MiTabla!N_PASAJE
'    App.HelpFile = "c:\hipertex\pasaje.hlp"
'End If
opF.Value = True'inicializa sin_contestar

Else

Set MiBase = OpenDatabase("c:\paquet\dbase\", False, True, "dbase IV;")
Set MiTabla = MiBase.OpenTable(Materia)
MiTabla.MoveFirst

'Label1.Caption = MiTabla!Pregunta
'label2.Caption = " " + MiTabla!A
'label3.Caption = " " + MiTabla!B
'label4.Caption = " " + MiTabla!C
'label5.Caption = " " + MiTabla!D

```

```
If MiTabla1Pregunta <> nul Then
    preguntas_x_seccion.Label1.Caption = MiTabla1Pregunta
Else
    preguntas_x_seccion.Label1.Caption = ""
End If
If MiTabla1A <> nul Then
    preguntas_x_seccion.opA = True
    preguntas_x_seccion.Label2.Caption = " " + MiTabla1A
Else
    preguntas_x_seccion.opA = False
    Label2.Caption = ""
End If
If MiTabla1B <> nul Then
    preguntas_x_seccion.opB = True
    preguntas_x_seccion.Label3.Caption = " " + MiTabla1B
Else
    preguntas_x_seccion.opB = False
    Label3.Caption = ""
End If
If MiTabla1C <> nul Then
    preguntas_x_seccion.opC = True
    preguntas_x_seccion.Label4.Caption = " " + MiTabla1C
Else
    preguntas_x_seccion.opC = False
    Label4.Caption = ""
End If
If MiTabla1D <> nul Then
    preguntas_x_seccion.opD = True
    preguntas_x_seccion.Label5.Caption = " " + MiTabla1D
Else
    preguntas_x_seccion.opD = False
    Label5.Caption = ""
End If
If MiTabla1E <> nul Then
    preguntas_x_seccion.opE = True
    preguntas_x_seccion.Label6.Caption = " " + MiTabla1E
Else
    preguntas_x_seccion.opE = False
    Label6.Caption = ""
End If

opF.Value = True'inicializa sin_contestar
End If
End Sub

Sub opA_Click (Value As Integer)
    Respuesta(i).R = "a"
    label8.Caption = "a"
End Sub

Sub opB_Click (Value As Integer)
    Respuesta(i).R = "b"
    label8.Caption = "b"
End Sub

Sub opC_Click (Value As Integer)
    Respuesta(i).R = "c"
    label8.Caption = "c"
```

FRAGUNIA.FRM - 0

End Sub

```
Sub opD_Click (Value As Integer)
    Respuesta(i).R = "d"
    label8.Caption = "d"
End Sub
```

```
Sub opE_Click (Value As Integer)
    Respuesta(i).R = "e"
    label8.Caption = "e"
End Sub
```

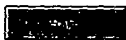
```
Sub pasaje_Click ()
    SendKeys "{f1}"
End Sub
```

```
Sub Timer1_Timer ()
    actual = Timer
    If actual > fin Then
        salvar.Show
    End If
    label11.Caption = " " + Time$
End Sub
```




Label10

Label11



MSV130K.FRM - 1

Sub Command3D2_Click ()

Open carga_calif.File1.FileName For Input As #1
Input #1, calexam.Pass, calexam.Autor, calexamtipo, calexam.Mater, calexam.Suspen, calexam.F

echa

```
Set MiBase = OpenDatabase("c:\paquet\dbase\*", False, True, "dbase IV;")
Set MiTabla = MiBase.OpenTable(calexam.Mater)
Close #1
Select Case LTrim(RTrim(calexam.Mater))
Case "Español"
    numero_a_sumar = 1000000
Case "literatu"
    numero_a_sumar = 2000000
'Case "Hist_univ"
'    numero_a_sumar = 3000000
Case "Hist_mex"
    numero_a_sumar = 4000000
Case "Geo_gral"
    numero_a_sumar = 5000000
Case "Geo_mex"
    numero_a_sumar = 6000000
'Case "matematicas"
'    numero_a_sumar = 7000000
'Case "fisica"
'    numero_a_sumar = 8000000
Case "Quimica"
    numero_a_sumar = 9000000
Case "biologia"
    numero_a_sumar = 0
End Select
```

```
MiTabla.Index = LTrim(RTrim(calexam.Mater))
labell1.Caption = Respuesta(s + 1).NoPreg
MiTabla.Seek "=", Respuesta(s + 1).NoPreg + numero_a_sumar
If MiTabla.Pregunta <> nul Then
```

```
    labell1.Caption = MiTabla.Pregunta
Else
    labell1.Caption = ""
End If
If MiTabla!A <> nul Then
    opAr.Visible = True
    Label2.Caption = " " + MiTabla!A
Else
    Label2.Caption = ""
    opAr.Visible = False
End If
If MiTabla!B <> nul Then
    opBr.Visible = True
    Label3.Caption = " " + MiTabla!B
Else
    Label3.Caption = ""
    opBr.Visible = False
End If
If MiTabla!C <> nul Then
    opCr.Visible = True
```

```

Label4.Caption = " " + MiTabla!C
Else
Label4.Caption = ""
opCr.Visible = False
End If
If MiTabla!D <> nul Then
opDr.Visible = True
Label5.Caption = " " + MiTabla!D
Else
Label5.Caption = ""
opDr.Visible = False
End If
If MiTabla!E <> nul Then
opEr.Visible = True
Label6.Caption = " " + MiTabla!E
Else
Label6.Caption = ""
opEr.Visible = False
End If
l = num_pregunta
num_pregunta = num_pregunta + 1
Label7.Caption = Str$(num_pregunta)
regresa = regresa + 1
If regresa < 0 Then
regresa = regresa + 1
End If
paloma.Left = 30
tache.Left = 30
paloma.Visible = False
tache.Visible = False
opAr.Visible = True
opBr.Visible = True
opCr.Visible = True
opDr.Visible = True
opEr.Visible = True
opFr.Value = True

If Respuesta(s + 1).N = Respuesta(s + 1).R Then
Select Case Respuesta(s + 1).H
Case "a"
opAr.Visible = False
paloma.Top = 2600
paloma.Visible = True
tache.Visible = False
Case "b"
opBr.Visible = False
paloma.Top = 3230
paloma.Visible = True
tache.Visible = False
Case "c"
opCr.Visible = False
paloma.Top = 3950
paloma.Visible = True
tache.Visible = False
Case "d"
opDr.Visible = False
paloma.Top = 4650
paloma.Visible = True
tache.Visible = False

```

RAVIBAK.FRM - 3

```
Case "e"
  opEr.Visible = False
  paloma.Top = 5380
  paloma.Visible = True
  tache.Visible = False
End Select
s = s + 1
Else
  Select Case Respuesta(s + 1).N
    Case "a"
      opAr.Visible = False
      paloma.Top = 2600
      paloma.Visible = True
    Case "b"
      opBr.Visible = False
      paloma.Top = 3230
      paloma.Visible = True
    Case "c"
      opCr.Visible = False
      paloma.Top = 3950
      paloma.Visible = True
    Case "d"
      opDr.Visible = False
      paloma.Top = 4660
      paloma.Visible = True
    Case "e"
      opEr.Visible = False
      paloma.Top = 5380
      paloma.Visible = True
  End Select
  Select Case Respuesta(s + 1).R
    Case "a"
      opAr.Visible = False
      tache.Top = 2600
      tache.Visible = True
    Case "b"
      opBr.Visible = False
      tache.Top = 3230
      tache.Visible = True
    Case "c"
      opCr.Visible = False
      tache.Top = 3950
      tache.Visible = True
    Case "d"
      opDr.Visible = False
      tache.Top = 4660
      tache.Visible = True
    Case "e"
      opEr.Visible = False
      tache.Top = 5380
      tache.Visible = True
  End Select
  s = s + 1
End If
End Sub

Sub Form_Load ()
  1 = 1
End Sub
```

KAVIADMM.FRM - 4

```
Open carga_calif.File1.FileName + "EXH" For Input As #1
Input #1, calexam.Pase, calexam.Autor, calexamtipo, calexam.Mater, calexam.Suspen, cale
xam.Fecha
Set MiBase = OpenDatabase("c:\paquet\dbase\*", False, True, "dbase IV;")
Set MiTabla = MiBase.OpenTable(calexam.Mater)
MiTabla.Index = ltrim(rtrim(calexam.Mater))
Close #1
Open calexam.Autor + ".CAL" For Input As #2
Input #2, cal(1).NoPregunta, cal(1).Respuesta, cal(1).Nota
Close #2
label1.Caption = cal(1).NoPregunta
MiTabla.Seek "=", cal(1).NoPregunta
```

```
Open carga_calif.File1.FileName For Input As #1
Input #1, calexam.Pase, calexam.Autor, calexamtipo, calexam.Mater, calexam.Suspen, cale
am.Fecha
Set MiBase = OpenDatabase("c:\paquet\dbase\*", False, True, "dbase IV;")
Set MiTabla = MiBase.OpenTable(calexam.Mater)
Close #1
MiTabla.Index = LTrim(RTrim(calexam.Mater))
Label9.Caption = Respuesta(1).NoPreg
Select Case calexam.Mater
Case "Espanol"
MiTabla.Seek "=", Respuesta(s + 1).NoPreg + 1000000
Case "Literatu"
MiTabla.Seek "=", Respuesta(s + 1).NoPreg + 2000000
Case "hist_univ"
MiTabla.Seek "=", Respuesta(s + 1).NoPreg + 3000000
Case "Hist_mex"
MiTabla.Seek "=", Respuesta(s + 1).NoPreg + 4000000
Case "Geo_gen"
MiTabla.Seek "=", Respuesta(s + 1).NoPreg + 5000000
Case "Geo_mex"
MiTabla.Seek "=", Respuesta(s + 1).NoPreg + 6000000
Case "matematicas"
MiTabla.Seek "=", Respuesta(s + 1).NoPreg + 7000000
Case "fisica"
MiTabla.Seek "=", Respuesta(s + 1).NoPreg + 8000000
Case "Quimica"
MiTabla.Seek "=", Respuesta(s + 1).NoPreg + 9000000
Case "Biologia"
MiTabla.Seek "=", Respuesta(s + 1).NoPreg + 0
End Select
If MiTabla!Pregunta <> nul Then
label1.Caption = MiTabla!Pregunta
Else
label1.Caption = ""
End If
If MiTabla!A <> nul Then
opAr.Visible = True
Label2.Caption = " " + MiTabla!A
Else
Label2.Caption = ""
opAr.Visible = False
End If
If MiTabla!B <> nul Then
opBr.Visible = True
```

```

Label3.Caption = " " + MiTabla1B
Else
Label3.Caption = ""
opBr.Visible = False
End If
If MiTabla1C <> nul Then
opCr.Visible = True
Label4.Caption = " " + MiTabla1C
Else
Label4.Caption = ""
opCr.Visible = False
End If
If MiTabla1D <> nul Then
opDr.Visible = True
Label5.Caption = " " + MiTabla1D
Else
Label5.Caption = ""
opDr.Visible = False
End If
If MiTabla1E <> nul Then
opEr.Visible = True
Label6.Caption = " " + MiTabla1E
Else
Label6.Caption = ""
opEr.Visible = False
End If

```

```

NumPregunta = 1
paloma.Left = 30
tache.Left = 30
paloma.Visible = False
tache.Visible = False
opAr.Visible = True
opBr.Visible = True
opCr.Visible = True
opDr.Visible = True
opEr.Visible = True
opFr.Value = True

```

```

If Respuesta(1).N = Respuesta(1).R Then
Select Case Respuesta(1).N
Case "a"
opAr.Visible = False
paloma.Top = 2600
paloma.Visible = True
tache.Visible = False
Case "b"
opBr.Visible = False
paloma.Top = 3230
paloma.Visible = True
tache.Visible = False
Case "c"
opCr.Visible = False
paloma.Top = 3950
paloma.Visible = True
tache.Visible = False
Case "d"

```

REVISION.FRM - 0

```

    opDr.Visible = False
    paloma.Top = 4660
    paloma.Visible = True
    tache.Visible = False
  Case "a"
    opPr.Visible = False
    paloma.Top = 5380
    paloma.Visible = True
    tache.Visible = False
  End Select
Else
  Select Case Respuesta(1).N
  Case "a"
    opAr.Visible = False
    paloma.Top = 2600
    paloma.Visible = True
  Case "b"
    opPr.Visible = False
    paloma.Top = 3230
    paloma.Visible = True
  Case "c"
    opCr.Visible = False
    paloma.Top = 3950
    paloma.Visible = True
  Case "d"
    opDr.Visible = False
    paloma.Top = 4660
    paloma.Visible = True
  Case "a"
    opPr.Visible = False
    paloma.Top = 5380
    paloma.Visible = True
  End Select
  Select Case Respuesta(1).R
  Case "a"
    opAr.Visible = False
    tache.Top = 2600
    tache.Visible = True
  Case "b"
    opPr.Visible = False
    tache.Top = 3230
    tache.Visible = True
  Case "c"
    opCr.Visible = False
    tache.Top = 3950
    tache.Visible = True
  Case "d"
    opDr.Visible = False
    tache.Top = 4660
    tache.Visible = True
  Case "a"
    opPr.Visible = False
    tache.Top = 5380
    tache.Visible = True
  End Select
End If
s = 1
End Sub
```

REVISAR.FRM - 1

```
Sub salida_Click ()
  Unload revisa
  s = 1
  num_pregunta = 1
End Sub
```


APENDICE "B"
MANUAL TÉCNICO

MANUAL TÉCNICO

OBJETIVO DEL MANUAL

La elaboración de éste manual, tiene como objetivo esencial el proporcionar soporte a los programadores para la adecuada operación del sistema "Tutor" tomando en cuenta las consideraciones de tipo técnico.

AMBIENTE DE OPERACION

Para la operación e instalación del sistema "Tutor" se debe contar con el siguiente ambiente computacional:

Un equipo de cómputo compatible IBM 286 o superior:

- Windows 3.1 instalado
- Mouse instalado.
- Monitor con tarjeta de video VGA, EGA.
- Un mínimo de 10 MBytes disponibles en disco duro.
- El sistema "Tutor" instalado.

La médula del desarrollo del sistema "Tutor" está basado en dos herramientas de Micro-Soft que son VisualBASIC y HC31. Estos dos programas conforman la infraestructura que hace posible la construcción del sistema.

DESCRIPCIÓN

Este manual contiene la información técnica requerida para tener una comprensión más amplia del sistema "Tutor" y se encuentra detallada en términos de como está constituida cada una de las partes que la integran. Se contemplan los aspectos necesarios para poder proporcionar mantenimiento al sistema y cuenta con una descripción de cada una de las formas y subrutinas (nombre y objetivo) que conforman y hacen posible su funcionamiento.

ACER.FRM [ACERCA DE]

SUB COMMAND1_CLICK()

MUESTRA ACER

AYUD.FRM [AYUDA]

SUB TEXT1_GOTFOCUS () + F1	ACCESA HIPERTEXTO BIOLOGÍA
SUB TEXT2_GOTFOCUS () + F1	ACCESA HIPERTEXTO ESPAÑOL
SUB TEXT3_GOTFOCUS () + F1	ACCESA HIPERTEXTO FÍSICA
SUB TEXT4_GOTFOCUS () + F1	ACCESA HIPERTEXTO GEOGRAFIA GENERAL
SUB TEXT5_GOTFOCUS () + F1	ACCESA HIPERTEXTO GEOGRAFÍA DE MÉXICO
SUB TEXT6_GOTFOCUS () + F1	ACCESA HIPERTEXTO HISTORIA DE MÉXICO
SUB TEXT7_GOTFOCUS () + F1	ACCESA HIPERTEXTO HISTORIA UNIVERSAL
SUB TEXT8_GOTFOCUS () + F1	ACCESA HIPERTEXTO LITERATURA
SUB TEXT9_GOTFOCUS () + F1	ACCESA HIPERTEXTO MATEMÁTICAS
SUB TEXT10_GOTFOCUS ()+ F1	ACCESA HIPERTEXTO QUÍMICA
SUB TEXT1_CLICK ()	ACCESA HIPERTEXTO BIOLOGIA
SUB TEXT2_CLICK ()	ACCESA HIPERTEXTO ESPAÑOL
SUB TEXT3_CLICK ()	ACCESA HIPERTEXTO FÍSICA
SUB TEXT4_CLICK ()	ACCESA HIPERTEXTO GEOGRAFIA GENERAL
SUB TEXT5_CLICK ()	ACCESA HIPERTEXTO GEOGRAFÍA DE MÉXICO
SUB TEXT6_CLICK ()	ACCESA HIPERTEXTO HISTORIA DE MÉXICO
SUB TEXT7_CLICK ()	ACCESA HIPERTEXTO HISTORIA UNIVERSAL
SUB TEXT8_CLICK ()	ACCESA HIPERTEXTO LITERATURA
SUB TEXT9_CLICK ()	ACCESA HIPERTEXTO MATEMÁTICAS
SUB TEXT10_CLICK ()	ACCESA HIPERTEXTO QUÍMICA
SUB COMMAND1_GOTFOCUS () +F1	ACCESA HIPERTEXTO "EL ESTUDIO DE LA BIOLOGIA"
SUB COMMAND2_GOTFOCUS () +F1	ACCESA HIPERTEXTO "ORGANIZACION BIOLOGICA "
SUB COMMAND3_GOTFOCUS () +F1	ACCESA HIPERTEXTO "SERES VIVOS Y LA ENERGIA "
SUB COMMAND4_GOTFOCUS () +F1	ACCESA HIPERTEXTO "REPRODUCCION "
SUB COMMAND5_GOTFOCUS () +F1	ACCESA HIPERTEXTO "GENETICA "
SUB COMMAND6_GOTFOCUS () +F1	ACCESA HIPERTEXTO "SALUD "

SUB COMMAND7_GOTFOCUS()+F1	ACCESA HIPERTEXTO "EVOLUCION "
SUB COMMAND8_GOTFOCUS()+F1	ACCESA HIPERTEXTO "MEDIO AMBIENTE "
SUB COMMAND9_GOTFOCUS()+F1	ACCESA HIPERTEXTO "COMUNIDADES BOTANICAS "
SUB COMMAND10_GOTFOCUS()+F1	ACCESA HIPERTEXTO "LINGUISTICA "
SUB COMMAND11_GOTFOCUS()+F1	ACCESA HIPERTEXTO "MORFOLOGIA "
SUB COMMAND12_GOTFOCUS()+F1	ACCESA HIPERTEXTO "SEMANTICA "
SUB COMMAND13_GOTFOCUS()+F1	ACCESA HIPERTEXTO "GRAMATICA "
SUB COMMAND14_GOTFOCUS()+F1	ACCESA HIPERTEXTO "EXPRESION ESCRITA "
SUB COMMAND15_GOTFOCUS()+F1	ACCESA HIPERTEXTO "LECTURA DE COMPRESION "
SUB COMMAND17_GOTFOCUS()+F1	ACCESA HIPERTEXTO "UNIDADES DE MEDICION "
SUB COMMAND18_GOTFOCUS()+F1	ACCESA HIPERTEXTO "MECANICA "
SUB COMMAND19_GOTFOCUS()+F1	ACCESA HIPERTEXTO "ONDAS "
SUB COMMAND20_GOTFOCUS()+F1	ACCESA HIPERTEXTO "MATERIA "
SUB COMMAND21_GOTFOCUS()+F1	ACCESA HIPERTEXTO "TERMODINAMICA "
SUB COMMAND22_GOTFOCUS()+F1	ACCESA HIPERTEXTO "ELECTRICIDAD "
SUB COMMAND23_GOTFOCUS()+F1	ACCESA HIPERTEXTO "ELECTROMAGNETISMO "
SUB COMMAND24_GOTFOCUS()+F1	ACCESA HIPERTEXTO "MEDIO AMBIENTE 2"
SUB COMMAND25_GOTFOCUS()+F1	ACCESA HIPERTEXTO "LA TIERRA "
SUB COMMAND26_GOTFOCUS()+F1	ACCESA HIPERTEXTO "LITOSFERA "
SUB COMMAND27_GOTFOCUS()+F1	ACCESA HIPERTEXTO "HIDROSFERA "
SUB COMMAND28_GOTFOCUS()+F1	ACCESA HIPERTEXTO "ATMOSFERA "
SUB COMMAND29_GOTFOCUS()+F1	ACCESA HIPERTEXTO "AMERICA "

SUB COMMAND30_GOTFOCUS() +F1	ACCESA HIPERTEXTO "EUROPA "
SUB COMMAND31_GOTFOCUS() +F1	ACCESA HIPERTEXTO "AFRICA "
SUB COMMAND32_GOTFOCUS() +F1	ACCESA HIPERTEXTO "ASIA "
SUB COMMAND33_GOTFOCUS() +F1	ACCESA HIPERTEXTO "SISTEMASECONOMICOS "
SUB COMMAND34_GOTFOCUS() +F1	ACCESA HIPERTEXTO "SITUACION DE MEXICO "
SUB COMMAND35_GOTFOCUS() +F1	ACCESA HIPERTEXTO "CARACTERISTICAS FISICAS "
SUB COMMAND36_GOTFOCUS() +F1	ACCESA HIPERTEXTO "POBLACION "
SUB COMMAND37_GOTFOCUS() +F1	ACCESA HIPERTEXTO "ZONAS ECONOMICAS "
SUB COMMAND38_GOTFOCUS() +F1	ACCESA HIPERTEXTO "MEXICO PREHISPANICO "
SUB COMMAND39_GOTFOCUS() +F1	ACCESA HIPERTEXTO "LA CONQUISTA "
SUB COMMAND40_GOTFOCUS() +F1	ACCESA HIPERTEXTO "LA COLONIA "
SUB COMMAND41_GOTFOCUS() +F1	ACCESA HIPERTEXTO "INDEPENDENCIA "
SUB COMMAND42_GOTFOCUS() +F1	ACCESA HIPERTEXTO "REVOLUCION "
SUB COMMAND43_GOTFOCUS() +F1	ACCESA HIPERTEXTO "MEXICO ACTUAL "
SUB COMMAND44_GOTFOCUS() +F1	ACCESA HIPERTEXTO "ESTUDIODELA HISTORIA "
SUB COMMAND45_GOTFOCUS() +F1	ACCESA HIPERTEXTO "CIVILIZACIONES "
SUB COMMAND46_GOTFOCUS() +F1	ACCESA HIPERTEXTO "EDAD MEDIA "
SUB COMMAND47_GOTFOCUS() +F1	ACCESA HIPERTEXTO "EL MUNDO MODERNO "
SUB COMMAND48_GOTFOCUS() +F1	ACCESA HIPERTEXTO "EL MUNDO CONTEMPORANEO "
SUB COMMAND49_GOTFOCUS() +F1	ACCESA HIPERTEXTO "EL MUNDO ACTUAL "
SUB COMMAND50_GOTFOCUS() +F1	ACCESA HIPERTEXTO "GENERALIDADES "
SUB COMMAND51_GOTFOCUS() +F1	ACCESA HIPERTEXTO "ANTIGUA "

SUB COMMAND52_GOTFOCUS()+F1	ACCESA HIPERTEXTO "CLASICA "
SUB COMMAND53_GOTFOCUS()+F1	ACCESA HIPERTEXTO "MEDIEVAL "
SUB COMMAND54_GOTFOCUS()+F1	ACCESA HIPERTEXTO "ESPAÑOLA "
SUB COMMAND55_GOTFOCUS()+F1	ACCESA HIPERTEXTO "PREHISPANICA "
SUB COMMAND56_GOTFOCUS()+F1	ACCESA HIPERTEXTO "BARROCA "
SUB COMMAND57_GOTFOCUS()+F1	ACCESA HIPERTEXTO "NEOCLASICA "
SUB COMMAND58_GOTFOCUS()+F1	ACCESA HIPERTEXTO "ROMANTICA "
SUB COMMAND59_GOTFOCUS()+F1	ACCESA HIPERTEXTO "REALISTA "
SUB COMMAND60_GOTFOCUS()+F1	ACCESA HIPERTEXTO "MODERNISTA "
SUB COMMAND61_GOTFOCUS()+F1	ACCESA HIPERTEXTO "NUMEROS NATURALES "
SUB COMMAND62_GOTFOCUS()+F1	ACCESA HIPERTEXTO "NUMEROS ENTEROS "
SUB COMMAND63_GOTFOCUS()+F1	ACCESA HIPERTEXTO "NUMEROS RACIONALES "
SUB COMMAND64_GOTFOCUS()+F1	ACCESA HIPERTEXTO "ALGEBRA "
SUB COMMAND65_GOTFOCUS()+F1	ACCESA HIPERTEXTO "GEOMETRIA "
SUB COMMAND66_GOTFOCUS()+F1	ACCESA HIPERTEXTO "TRIGONOMETRIA "
SUB COMMAND67_GOTFOCUS()+F1	ACCESA HIPERTEXTO "INTRODUCCION "
SUB COMMAND68_GOTFOCUS()+F1	ACCESA HIPERTEXTO "EL ATOMO "
SUB COMMAND69_GOTFOCUS()+F1	ACCESA HIPERTEXTO "LOS ELEMENTOS "
SUB COMMAND70_GOTFOCUS()+F1	ACCESA HIPERTEXTO "ENLACES "
SUB COMMAND71_GOTFOCUS()+F1	ACCESA HIPERTEXTO "ACIDOS Y BASES "
SUB COMMAND72_GOTFOCUS()+F1	ACCESA HIPERTEXTO "CALCULOS QUIMICOS "
SUB COMMAND73_GOTFOCUS()+F1	ACCESA HIPERTEXTO "REACCIONES QUIMICAS "

SUB COMMAND1_CLICK ()	ACCESA HIPERTEXTO "EL ESTUDIO DE LA BIOLOGIA "
SUB COMMAND2_CLICK ()	ACCESA HIPERTEXTO "ORGANIZACION BIOLOGICA "
SUB COMMAND3_CLICK ()	ACCESA HIPERTEXTO "SERES VIVOS Y LA ENERGIA "
SUB COMMAND4_CLICK ()	ACCESA HIPERTEXTO "REPRODUCCION "
SUB COMMAND5_CLICK ()	ACCESA HIPERTEXTO "GENETICA "
SUB COMMAND6_CLICK ()	ACCESA HIPERTEXTO "SALUD "
SUB COMMAND7_CLICK ()	ACCESA HIPERTEXTO "EVOLUCION "
SUB COMMAND8_CLICK ()	ACCESA HIPERTEXTO "MEDIO AMBIENTE "
SUB COMMAND9_CLICK ()	ACCESA HIPERTEXTO "COMUNIDADES BOTANICAS "
SUB COMMAND10_CLICK ()	ACCESA HIPERTEXTO "LINGUISTICA "
SUB COMMAND11_CLICK ()	ACCESA HIPERTEXTO "MORFOLOGIA "
SUB COMMAND12_CLICK ()	ACCESA HIPERTEXTO "SEMANTICA "
SUB COMMAND13_CLICK ()	ACCESA HIPERTEXTO "GRAMATICA "
SUB COMMAND14_CLICK ()	ACCESA HIPERTEXTO "EXPRESION ESCRITA "
SUB COMMAND15_CLICK ()	ACCESA HIPERTEXTO "LECURA DE COMPRESION "
SUB COMMAND16_CLICK ()	ACCESA HIPERTEXTO "LECTURA DE COMPRESION "
SUB COMMAND17_CLICK ()	ACCESA HIPERTEXTO "UNIDADES DE MEDICION "
SUB COMMAND18_CLICK ()	ACCESA HIPERTEXTO "MECANICA "
SUB COMMAND19_CLICK ()	ACCESA HIPERTEXTO "ONDAS "
SUB COMMAND20_CLICK ()	ACCESA HIPERTEXTO "MATERIA "
SUB COMMAND21_CLICK ()	ACCESA HIPERTEXTO "TERMODINAMICA "
SUB COMMAND22_CLICK ()	ACCESA HIPERTEXTO "ELECTRICIDAD "
SUB COMMAND23_CLICK ()	ACCESA HIPERTEXTO "ELECTROMAGNETISMO "
SUB COMMAND24_CLICK ()	ACCESA HIPERTEXTO "MEDIO AMBIENTE 2"
SUB COMMAND25_CLICK ()	ACCESA HIPERTEXTO "LA TIERRA "
SUB COMMAND26_CLICK ()	ACCESA HIPERTEXTO "LITOSFERA "
SUB COMMAND27_CLICK ()	ACCESA HIPERTEXTO "HIDROSFERA "
SUB COMMAND28_CLICK ()	ACCESA HIPERTEXTO "ATMOSFERA "
SUB COMMAND29_CLICK ()	ACCESA HIPERTEXTO "AMERICA "
SUB COMMAND30_CLICK ()	ACCESA HIPERTEXTO "EUROPA "
SUB COMMAND31_CLICK ()	ACCESA HIPERTEXTO "AFRICA "

SUB COMMAND32_CLICK()	ACCESA HIPERTEXTO "ASIA "
SUB COMMAND33_CLICK()	ACCESA HIPERTEXTO "SISTEMASECONOMICOS "
SUB COMMAND34_CLICK()	ACCESA HIPERTEXTO "SITUACION DE MEXICO "
SUB COMMAND35_CLICK()	ACCESA HIPERTEXTO "CARACTERISTICAS FISICAS "
SUB COMMAND36_CLICK()	ACCESA HIPERTEXTO "POBLACION "
SUB COMMAND37_CLICK()	ACCESA HIPERTEXTO "ZONAS ECONOMICAS "
SUB COMMAND38_CLICK()	ACCESA HIPERTEXTO "MEXICO PREHISPANICO "
SUB COMMAND39_CLICK()	ACCESA HIPERTEXTO "LA CONQUISTA "
SUB COMMAND40_CLICK()	ACCESA HIPERTEXTO "LA COLONIA "
SUB COMMAND41_CLICK()	ACCESA HIPERTEXTO "INDEPENDENCIA "
SUB COMMAND42_CLICK()	ACCESA HIPERTEXTO "REVOLUCION "
SUB COMMAND43_CLICK()	ACCESA HIPERTEXTO "MEXICO ACTUAL "
SUB COMMAND44_CLICK()	ACCESA HIPERTEXTO "ESTUDIODELA HISTORIA "
SUB COMMAND45_CLICK()	ACCESA HIPERTEXTO "CIVILIZACIONES "
SUB COMMAND46_CLICK()	ACCESA HIPERTEXTO "EDAD MEDIA "
SUB COMMAND47_CLICK()	ACCESA HIPERTEXTO "EL MUNDO MODERNO "
SUB COMMAND48_CLICK()	ACCESA HIPERTEXTO "EL MUNDO CONTEMPORANEO "
SUB COMMAND49_CLICK()	ACCESA HIPERTEXTO "EL MUNDO ACTUAL "
SUB COMMAND50_CLICK()	ACCESA HIPERTEXTO "GENERALIDADES "
SUB COMMAND51_CLICK()	ACCESA HIPERTEXTO "ANTIGUA "
SUB COMMAND52_CLICK()	ACCESA HIPERTEXTO "CLASICA "
SUB COMMAND53_CLICK()	ACCESA HIPERTEXTO "MEDIEVAL "
SUB COMMAND54_CLICK()	ACCESA HIPERTEXTO "ESPANOLA "
SUB COMMAND55_CLICK()	ACCESA HIPERTEXTO "PREHISPANICA "
SUB COMMAND56_CLICK()	ACCESA HIPERTEXTO "BARROCA "
SUB COMMAND57_CLICK()	ACCESA HIPERTEXTO "NEOCLASICA "
SUB COMMAND58_CLICK()	ACCESA HIPERTEXTO "ROMANTICA "
SUB COMMAND59_CLICK()	ACCESA HIPERTEXTO "REALISTA "
SUB COMMAND60_CLICK()	ACCESA HIPERTEXTO "MODERNISTA "
SUB COMMAND61_CLICK()	ACCESA HIPERTEXTO "NUMEROS NATURALES "

SUB COMMAND62_CLICK ()	ACCESA HIPERTEXTO "NUMEROS ENTEROS "
SUB COMMAND63_CLICK ()	ACCESA HIPERTEXTO "NUMEROS RACIONALES "
SUB COMMAND64_CLICK ()	ACCESA HIPERTEXTO "ALGEBRA "
SUB COMMAND65_CLICK ()	ACCESA HIPERTEXTO "GEOMETRIA "
SUB COMMAND66_CLICK ()	ACCESA HIPERTEXTO "TRIGONOMETRIA "
SUB COMMAND67_CLICK ()	ACCESA HIPERTEXTO "INTRODUCCION "
SUB COMMAND68_CLICK ()	ACCESA HIPERTEXTO "EL ATOMO "
SUB COMMAND69_CLICK ()	ACCESA HIPERTEXTO "LOS ELEMENTOS "
SUB COMMAND70_CLICK ()	ACCESA HIPERTEXTO "ENLACES "
SUB COMMAND71_CLICK ()	ACCESA HIPERTEXTO "ACIDOS Y BASES "
SUB COMMAND72_CLICK ()	ACCESA HIPERTEXTO "CALCULOS QUIMICOS "
SUB COMMAND73_CLICK ()	ACCESA HIPERTEXTO "REACCIONES QUIMICAS "

CALCULAD.FRM [CALCULADORA]

SUB BORRAENTRADA_CLICK ()	BORRA LA ULTIMA ENTRADA "CE"
SUB DIGITO_CLICK (INDEX AS INTEGER)	CONCATENA LAS ENTRADAS DE LOS DIGITOS Y PUNTO DECIMAL DE LA CALCULADORA
SUB FORM_LOAD ()	INICIALIZA LOS OPERANDOS Y LA ULTIMA ENTRADA
SUB INICIALIZA_CLICK ()	INICIALIZA LA PANTALLA A CERO Y EJECUTA FORM_LOAD
SUB OPERACION_CLICK ()	SELECCIONA LA OPERACION A REALIZAR
SUB PUNTO_DEC_CLICK ()	CONCATENA EL PUNTO DECIMAL
SUB TANTO:PORCIENTO_CLICK ()	REALIZA LAS OPERACIONES DE PORCENTAJE

CARGA_SU.FRM [CARGA EXAMENES SUSPENDIDOS]

SUB FILE1_CLICK ()	ABRE LOS ARCHIVOS DE UN EXAMEN SUSPENDIDO
SUB FILE1_Db1Click ()	CARGA LAS RESPUESTAS Y NUMEROS DE PREGUNTAS DEL EXAMEN SELECCIONADO

EJECUTAR.FRM [REALIZA UN ESCAPE A DOS]

SUB COMMAND2_CLICK ()	INICIALIZA EL COMBOBOX
SUB ACEPTAR_CLICK ()	ESCAPE A SHELL
SUB CANCELAR_CLICK ()	CANCELA LA ACCION DE EJECUTAR PROCESO
SUB COMBO1_KEYPRESS(KEYASCII AS INTEGER)	CHECA EL VALOR DE MAYUSCULAS
SUB MAYUSCULAS_CLICK ()	CAMBIA A MAYUSCULAS

EXAMEN.FRM [REALIZA EXAMENES]

SUB OPTION3D1(VALUE AS INTEGER)	ASIGNA LA RESPUESTA "a"
SUB OPTION3D2(VALUE AS INTEGER)	ASIGNA LA RESPUESTA "b"
SUB OPTION3D3(VALUE AS INTEGER)	ASIGNA LA RESPUESTA "c"
SUB OPTION3D4(VALUE AS INTEGER)	ASIGNA LA RESPUESTA "d"
SUB OPTION3D5(VALUE AS INTEGER)	ASIGNA LA RESPUESTA "e"
SUB OPTION3D6(VALUE AS INTEGER)	ASIGNA LA RESPUESTA "f"
SUB CANCELAR_CLICK ()	DESHABILITA CONFIRMAR Y CANCELAR
SUB COMMAND3D1_CLICK ()	GENERA PREGUNTA ALEATORIA HACIA ADELANTE
SUB COMMAND3D2_CLICK ()	GENERA PREGUNTAS HACIA ATRAS
SUB COMMAND3D3_CLICK ()	HABILITA CONFIRMAR Y CANCELAR
SUB CONFIRMAR_CLICK ()	MUESTRA SALVAR
SUB FORM_LOAD ()	ABRE LA BASE DE DATOS
SUB PASAJE_CLICK ()	ACTIVA EL PASAJE
SUB AYUDA_CLICK ()	ACCESA LA AYUDA
SUB TIMER1_TIMER ()	SI EXPIRA EL TIEMPO SUSPENDE EL EXAMEN Y MUESTRA SALVAR

EXAMEN_X.FRM [SELECCIONA EXAMEN POR SECCION]

SUB COMMAND3D1_CLICK ()	MUESTRA PORTADA E INICIALIZA UN EXAMEN DE BIOLOGIA
SUB COMMAND3D2_CLICK ()	MUESTRA PORTADA E INICIALIZA UN EXAMEN DE ESPAÑOL
SUB COMMAND3D3_CLICK ()	MUESTRA PORTADA E INICIALIZA UN EXAMEN DE FISICA

SUB COMMAND3D4_CLICK ()	MUESTRA PORTADA E INICIALIZA UN EXAMEN DE GEOGRAFIA GENERAL
SUB COMMAND3D5_CLICK ()	MUESTRA PORTADA E INICIALIZA UN EXAMEN DE GEOGRAFIA DE MEXICO
SUB COMMAND3D6_CLICK ()	MUESTRA PORTADA E INICIALIZA UN EXAMEN DE HISTORIA DE MEXICO
SUB COMMAND3D7_CLICK ()	MUESTRA PORTADA E INICIALIZA UN EXAMEN DE HISTORIA UNIVERSAL
SUB COMMAND3D8_CLICK ()	MUESTRA PORTADA E INICIALIZA UN EXAMEN DE LITERATURA
SUB COMMAND3D9_CLICK ()	MUESTRA PORTADA E INICIALIZA UN EXAMEN DE MATEMATICAS
SUB COMMAND3D10_CLICK ()	MUESTRA PORTADA E INICIALIZA UN EXAMEN DE QUIMICA
SUB COMMAND3D1_GOTFOCUS()+F1	ACCESA AYUDA DE BIOLOGIA
SUB COMMAND3D2_GOTFOCUS()+F1	ACCESA AYUDA DE ESPAÑOL
SUB COMMAND3D3_GOTFOCUS ()+F1	ACCESA AYUDA DE FISICA
SUB COMMAND3D4_GOTFOCUS ()+F1	ACCESA AYUDA DE GEOGRAFIA GENERAL
SUB COMMAND3D5_GOTFOCUS ()+F1	ACCESA AYUDA DE GEOGRAFIA DE MEXICO
SUB COMMAND3D6_GOTFOCUS ()+F1	ACCESA AYUDA DE HISTORIA DE MEXICO
SUB COMMAND3D7_GOTFOCUS ()+F1	ACCESA AYUDA DE HISTORIA UNIVERSAL
SUB COMMAND3D8_GOTFOCUS ()+F1	ACCESA AYUDA DE LITERATURA
SUB COMMAND3D9_GOTFOCUS ()+F1	ACCESA AYUDA DE MATEMATICAS
SUB COMMAND3D10_GOTFOCUS ()+F1	ACCESA AYUDA DE QUIMICA
SUB COMMAND11_CLICK	ESCONDE EXAMEN_X_SECCION

SALVAR.FRM [GENERA ARCHIVOS DE EXAMENES SUSPENDIDOS O TERMINADOS]

SUB ACEPTAR_CLICK ()	SALVA LA INFORMACION DE UN EXAMEN
SUB CANCELAR_CLICK ()	ESCONDE SALVAR
SUB FORM_LOAD ()	SELECCIONA EL TIPO DE EXAMEN

TIPO_EXAMEN.FRM [SELECCIONA TIPO DE EXAMEN]

SUB GLOBAL_CLICK ()	INICIALIZA EXAMEN GLOBAL
SUB SECCION_CLICK ()	INICIALIZA EXAMEN POR SECCION
SUB EXAMEN_GLOBAL_CLICK ()	MUESTRA PREGUNTAS GLOBAL
SUB EXAMEN_SECCION_CLICK ()	MUESTRA PREGUNTAS_X_SECCION

TUTOR_O.FRM [SELECCIONA EL MODO DEL EXAMEN POR SECCION]

SUB EXAMEN_EXAMEN_CLICK ()	MODO EXAMEN Y MUESTRA PREGUNTAS_X_SECCION
SUB EXAMEN_TUTOR_CLICK ()	MODO TUTOR Y MUESTRA PREGUNTAS_GAUGE

REVISAR.FRM [CALIFICA EXAMENES]

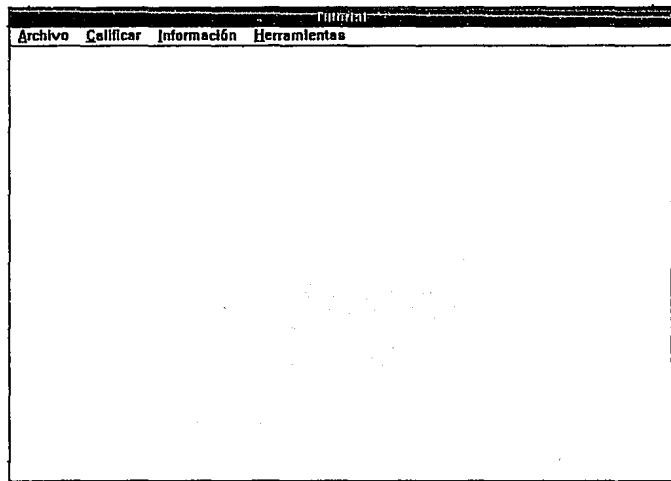
SUB COMMAND3D1_CLICK ()	ABRE LOS ARCHIVOS CORRESPONDIENTES, LAS BASES DE DATOS Y CARGA LAS PREGUNTAS, COMPARA LAS RESPUESTAS DEL USUARIO CON LAS DE LAS BASES DE DATOS Y CALIFICA
SUB FORM_LOAD ()	LANZA LA 1ª PREGUNTA
SUB SALIDA_CLICK	QUITA, REvisa E INICIALIZA LOS CONTADORES DE PREGUNTA.

CARTA DE NAVEGACIÓN

APENDICE "C"
MANUAL DE USUARIO

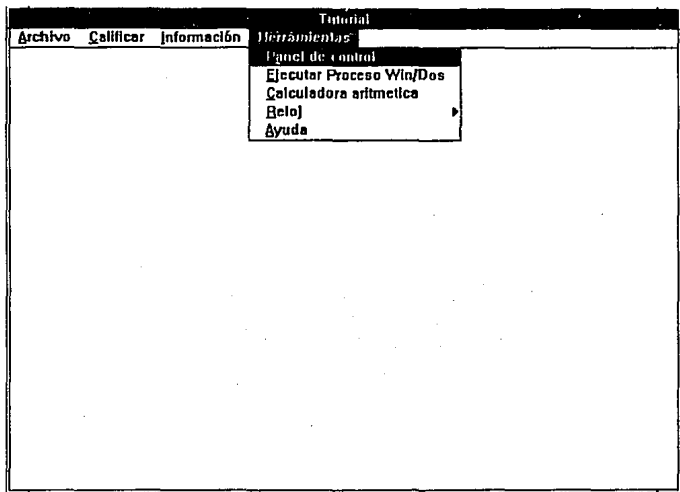
MANUAL DE USUARIO

El manual del usuario del sistema tutorial se encuentra organizado en forma jerárquica, partiendo desde el menú principal hasta cada una de las opciones y herramientas disponibles. El manejo del tutorial es completamente bajo filosofía Windows; es decir, que para aquellos usuarios que estén completamente familiarizados con este estilo de paquetería, el manejo del paquete será una tarea fácil de aprender y dominar.



Esta pantalla es el esquema del menú principal. Las cuatro opciones que pueden apreciarse se accesan con tres diferentes opciones:

1. Presionando la tecla ALT, automáticamente la primera opción de nuestro menú queda preseleccionada (la opción queda sombreada en tono azul); y, posteriormente presionando la tecla ENTER la opción es seleccionada. Para desplazarse al resto de las opciones del menú se hace uso de las teclas de desplazamiento (flechas).
2. Directamente se selecciona la opción posicionándose con el mouse.
3. Presionando las teclas ALT + Primera letra de la opción deseada.



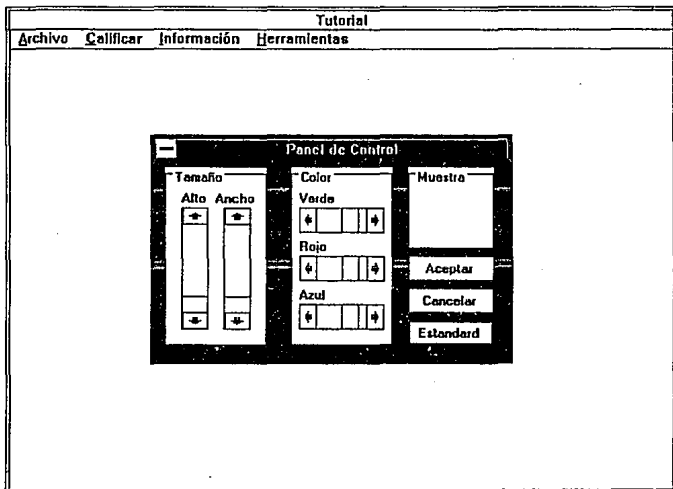
Alternativas para la opción Herramientas.

La ejecución de procesos proporciona la facilidad de ejecutar archivos EXE, COM y BAT.

La calculadora realiza operaciones aritméticas.

El Reloj visualiza un reloj analógico o digital.

Ayuda nos proporciona el sistema de hipertexto asociado al " Tutor "



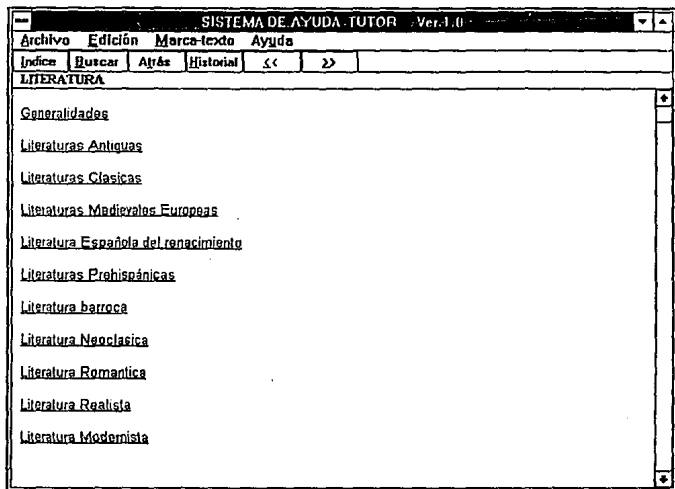
El Panel de Control nos da la facilidad de cambiar las características de la pantalla principal pudiendo modificar el tamaño y color del panel, o en su caso, cancelar quedando vigentes las características actuales.

Ayuda

Biología	Geografía general	Historia universal	Química
El estudio de la biología	Medio ambiente	Estudio de la historia	Introducción
Organización biológica	La tierra	Civilizaciones	El átomo
Seres vivos y la energía	Litósfera	Edad media	Los elementos
Reproducción	Hidrosfera	El mundo moderno	Enlaces
Genética	Atmósfera	El mundo contemporáneo	Ácidos y bases
Salud	América	El mundo actual	Cálculos químicos
Evolución	Europa		Reacciones químicas
Medio ambiente	África	Literatura	
Comunidades botánicas	Asia	Generalidades	
	Sistemas económicos	Antigua	
Español	Geografía de México	Clásica	
Lingüística	Situación de México	Medieval Europea	
Morfología	Caract. Físicas	Española renacentista	
Semántica	Población	Prehispánica	
Gramática	Zonas económicas	Barroca	
Expresión escrita		Neoclásica	
Lectura de comprensión		Romántica	
		Realista	
		Modernista	
Física	Historia de México	Matemáticas	
Unidades de medición	México prehispánico	Números naturales	
Mecánica	La conquista	Números enteros	
Ondas	La colonia	Números racionales	
Materia	Independencia	Álgebra	
Termodinámica	Revolución	Geometría	
Electricidad	México actual	Trigonometría	
Electromagnetismo			

Salir

Al seleccionar Ayuda aparece un menú con las áreas de estudio que comprende el tutorial así como los temas que comprende cada una de ellas.

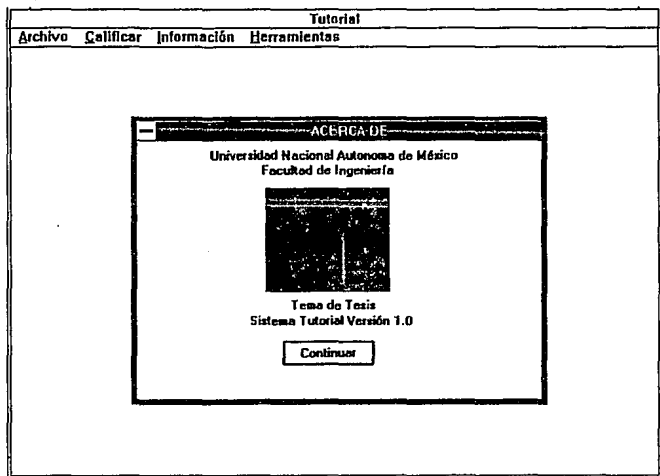


Esta pantalla con las opciones que se aprecian en la parte superior constituyen el control para los textos de ayuda.

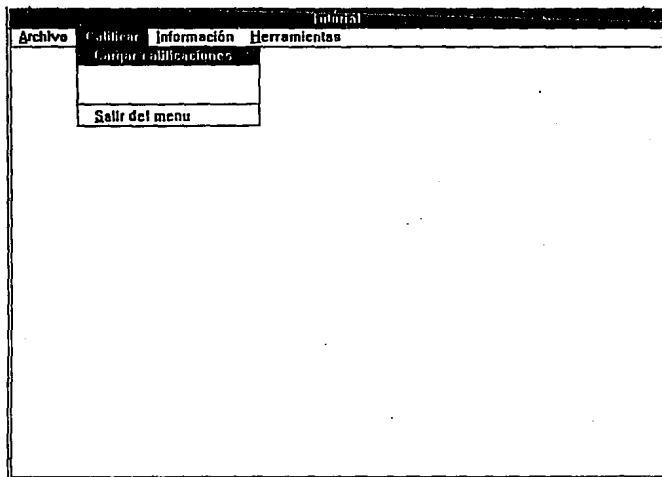
La descripción de los botones siguientes son para la manipulación del texto consultado.

1. **Índice:** Dependiendo del texto de ayuda que haya sido seleccionado, al elegir esta opción, el control del help se regresa directamente al índice de materia correspondiente.
2. **Buscar:** Se tiene la facilidad de hacer búsquedas de definiciones, conceptos, etc. relacionados al tema en consulta.
3. **Atrás:** Automáticamente devuelve el control al paso anterior realizado. Por ejemplo, si nos encontramos parados en algún tema de matemáticas; nuestro paso previo anterior sería el índice de matemáticas.
4. **Historial:** Lleva una secuencia de los temas, subtemas que han sido consultados.

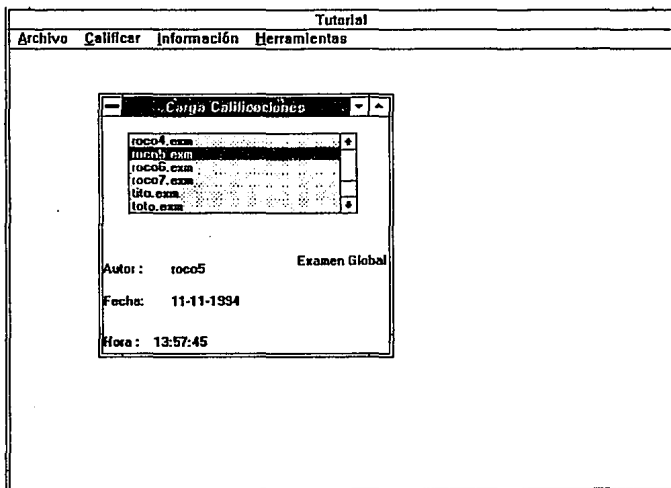
5. **Botones de avance y retroceso:** Permiten navegar hacia atrás o adelante dentro de los textos de ayuda.
6. **Menú Windows(Archivo, Edicion, Marca_Texto, Ayuda):** Permite imprimir un tema, salir, y diversas operaciones relacionadas al menú de ayuda, además del sistema de "ayuda para usar la ayuda"(valga la redundancia).



Pantalla correspondiente a la opción de menú **información**. Número de versión e Institución a la que pertenece el tutorial.



Cargar calificaciones: Permite cargar un examen calificado en caso de existir alguno; proporcionando información acerca del examen (autor, fecha de realización, hora y tipo de examen).



Selecciona un archivo para calificar, con "doble click" revisa calificaciones.

Revisa Examen por Sección

Si por una resistencia fluye una corriente de 7 amperes al aplicarse una diferencia de potencial de 28 volts, el valor de la resistencia es:

0.250 Ohms
 0.500 Ohms
 3 Ohms
 5 Ohms
 4 Ohms

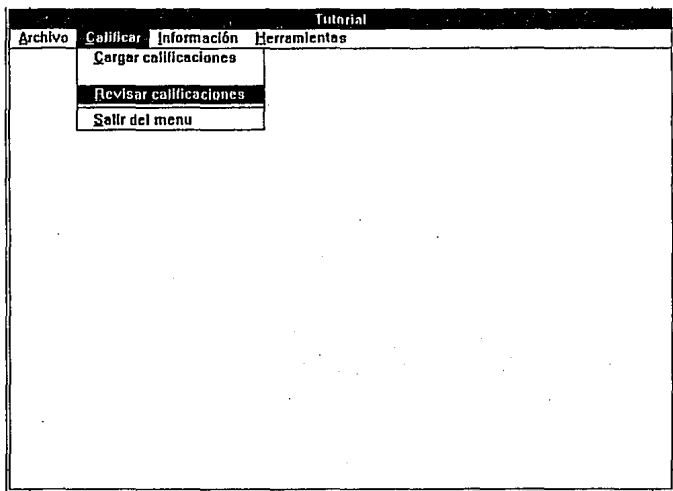
Ítem	
1	a
2	b
3	d
4	f
5	j
6	k
7	l
8	m
9	n
10	p
11	q

En el menú revisar:

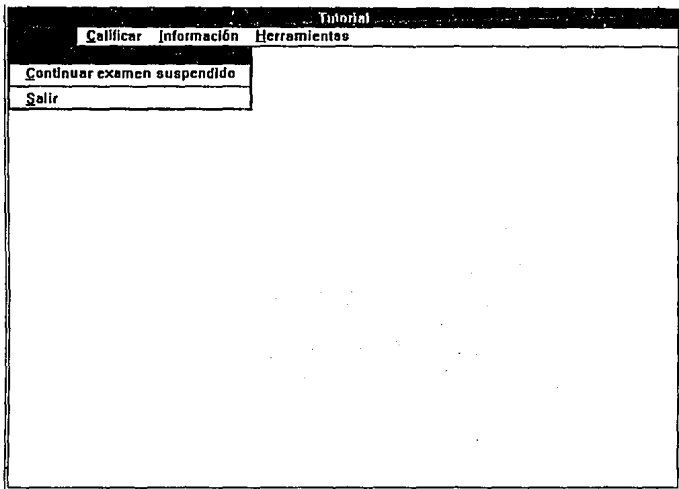
Las palomas en azul, indican la respuesta correcta. En su caso, las respuestas marcadas con tache en rojo indican que la pregunta fue contestada erróneamente.

Los iconos corresponden a:

- Libro: proporciona/esconde botón ayuda.
- Reloj: proporciona/esconde la hora del sistema.
- Lentes: proporciona/esconde botón de pasaje, en las materias donde se requiere pasaje (fabulas, textos, figuras, etc) relacionados a las preguntas, asocian el hipertexto correspondiente.
- Calabera: proporciona/esconde botón de suspender examen.
- Lista: despliega y repliega la lista de preguntas para trabajar de una manera más cómoda.



Revisar Calificaciones: Es válida esta opción si y solo si, previamente se ha cargado algún examen existente.

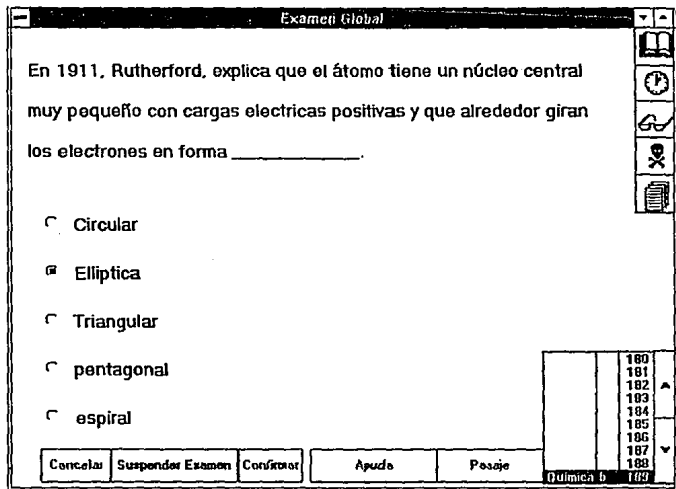


Comenzar un nuevo examen: Ofrece dos alternativas.

Examen global: Comprende un porcentaje definido de preguntas por cada una de las materias comprendidas por el tutorial.

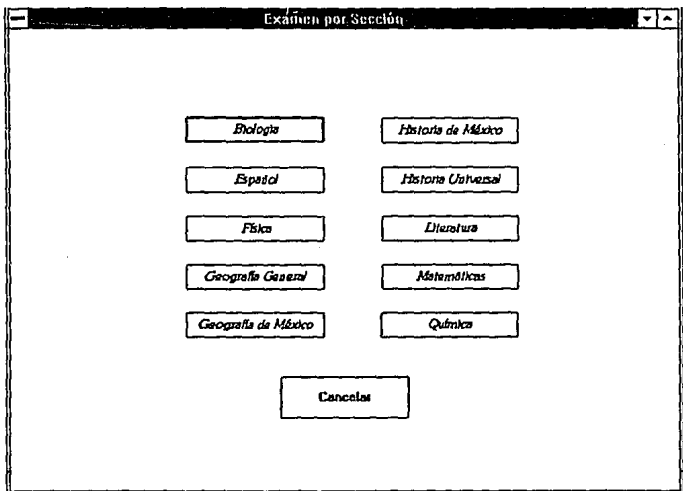
Examen por sección: Da la alternativa de hacer un examen específico de alguna materia ya sea en modo tutor o modo examen.

En cualquiera de los dos tipos de examen, al terminar de contestar las preguntas se podrá salvar el examen si así es deseado.

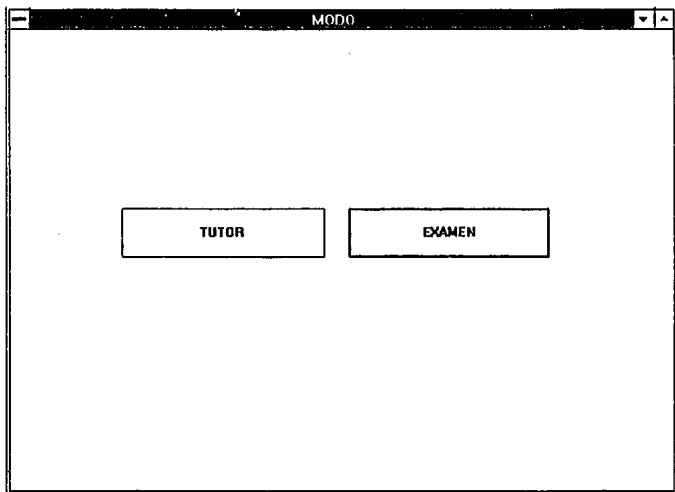


Examen Global: Al realizar esta selección, se trabaja bajo la filosofía mostrada. Se deben recordar los siguientes puntos:

1. Siempre que sea válido, podrá confirmarse o cancelarse la suspensión de un examen.
2. Los botones de ayuda y pasaje estarán disponibles siempre y cuando no estén en tono gris.
3. Puede tenerse un desplazamiento libre de avance y retroceso dentro del examen en donde puede visualizar se el número de pregunta y materia, lo cual permite contestar y trabajar más ágilmente.
4. La realización del examen global dura como máximo tres horas y media y consta de 300 preguntas.



Pantalla donde se selecciona Examen por sección.



Quando se selecciona Examen por sección, existen el modo tutor y modo examen.

Preguntas					8 de 30	
La atmosfera esta constituida por Nitrogeno 78% ; Oxigeno 21% ; Dioxido de carbono 0.04% y _____ 0.96% .						
<input type="radio"/> Gases nobles						
<input type="radio"/> Electroliticos						
<input checked="" type="radio"/> Gases Raros						
<input type="radio"/> Gases puros						
<input type="radio"/> Gases toxicos						
Cancelar	Suspender Tutor	Confirmar	Ayuda	Pasejo	>>	

En modo tutor, se hace un examen que consiste de 30 preguntas con 20 minutos como límite, con opción de avance ascendente únicamente. Se aprecia en la parte superior derecha de la pantalla un indicador de aciertos en tono rojo "gauge" que mide el avance del usuario.

Examen por sección

¿Cual es el radical nitrato ?






OH

SO₃

NO₂

ClC

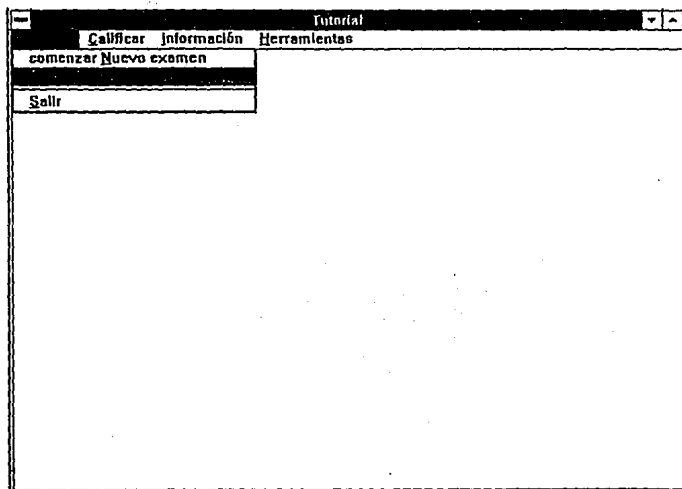
SO₂

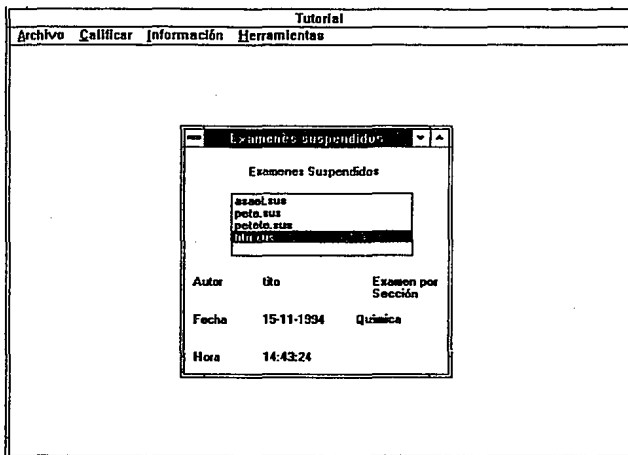
1		
2		
3		
4		
5		
6		
7		
8		
9		
10		
11		

El modo examen se realiza y maneja de manera similar a examen global, solo que en este caso nos limitamos a la materia que fue seleccionada.

Consta de 30 preguntas y dura 20 minutos.



Continuar examen suspendido permite terminar de contestar exámenes que ya habían sido iniciados.



Continuar exámenes suspendidos. Permite cargar un examen suspendido, con "doble click " continua el examen correspondiente.