2
2e)

UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO

FACULTAD DE CONTADURIA Y ADMINISTRACION

LOGMAT

Sistema tutorial para la enseñanza-aprendizaje de Lógica Matemática.

Seminario de Investigación Informática

que en opción al grado de

Licenciada en Informática

P R E S E N T A

MARIBEL ASENCIO ARMENTA

Director del Seminario: C.P. y L.A. José Antonio Echenique García

CALIDAD ACADEMICA

FCA

MEXICO, D. F.

1994

UNAM – Dirección General de Bibliotecas Tesis Digitales Restricciones de uso

DEDICATORIA

A mis a padres:

Sr. Juan A. Asencio Loredo y **Sra. Rosa Armenta de Asencio** con quienes comparto la cristalización de este anhelo. En especial a mi padre que ha sido el mayor apoyo a lo largo de mis estudios y constante ejemplo de integridad, disciplina, y perseverancia.

A mis hermanos:

Georgina, Juan Antonio y **Erika** por el gran afecto que les tengo, deseo que logren sus más preciadas metas y tengan éxito en la vida.

A mis tías:

Sra. Estela Asencio Loredo y **Sra. Gloria Armenta Zepeda** por guiarme y tenderme siempre la mano.

AGRADECIMIENTOS

Ayer, hoy y siempre a nuestra Alma Mater, **UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO.**

Mi enorme cariño y gratitud a la inolvidable **FACULTAD DE CONTADURIA Y ADMINISTRACION,** inquebrantable fortaleza de educación, conocimiento y progreso; es un orgullo pertenecer a ella.

Al **C.P. y L.A. José Antonio Echenique García,** quien ha impreso en los lugares que ocupa dentro de nuestra Facultad, su sello inconfundible de altruismo, profesionalismo y superación, lo que convierte a su orientación personal y académica en estímulos para resolver nuestras dificultades como estudiantes.

A la **Lic. Ana Hilda Gómez Torres,** cuya actitud en el asesoramiento del presente seminario se expresó desde el comienzo de él en una dedicación y atención superiores al compromiso académico. La satisfacción de haber culminado el trabajo intensivo al que nos sometimos se debe en gran medida a su experiencia profesional, acuciosa lectura de borradores y texto final, así como a sus consejos y apoyo en pro de mi preparación.

Al **Lic. Ricardo Alfredo Varela Juárez** por su estimulante apoyo, trato cordial y amistoso.

Al **Lic. Alfredo Aguirre Piña** y a la **Lic. Lourdes Arroyo Shuartez** por sus alentadoras palabras.

Al **Departamento de Matemáticas de la Facultad de Contaduría y Administración, Mat. Gilberto Prieto Morín** y **Sr. José Luis García León,** por su valiosa ayuda, participación, confianza y todas las facilidades brindadas en la elaboración del presente trabajo.

Al **Lic. Luis Eduardo López Castro** y al **Mat. José Alfredo Amor Montaño** por su receptividad a las consultas y sus atinadas sugerencias que contribuyeron al mejoramiento de este trabajo.

A mis compañeros y amigos, **Verónica de la Rosa Guerrero, Rosa Torres Gálvez, Norma Carrasco Martínez, Hector Ruíz García, Norma Elvira Peralta Márquez, Ana Luisa Prieto Espinosa y Antonio Aranda Román.**

Deseo también hacer patente mi agradecimiento a todas aquellas personas que colaboraron en la etapa de evaluación de este trabajo.

Pensar en computadoras aplicadas a la enseñanza no significa pensar en las computadoras, significa pensar en la enseñanza.

Alan Ellis, 1974.

I N D I C E

INTRODUCCION.

La Facultad de Contaduría y Administración (FCA) de la Universidad Nacional Autónoma de México (UNAM) tiene como misión: formar integralmente profesionales y personas con grado académico útiles a la sociedad que, con base en sus respectivos marcos teóricos, sean capaces de analizar problemas y proponer soluciones dentro de los más altos valores éticos, sociales y culturales. [1]

Desde 1985 los planes de estudio vigentes fueron diseñados en respuesta a las necesidades que la sociedad planteaba en ese momento. Sin embargo, el desarrollo económico, la política económica y la tecnología disponible en nuestro país han cambiado en los últimos años. Estas nuevas condiciones hacen que las necesidades y expectativas de las organizaciones en relación con el desempeño de los egresados de la FCA sean modificados en 1993.

Las modificaciones a los planes de estudio 1993 obedecen a la necesidad de fortalecer la preparación académica de los estudiantes para que respondan con suficiencia a los requerimientos actuales y futuros de la sociedad.

Con fundamento en lo anterior, se pensó en la sugerencia de un nuevo método didáctico como apoyo educativo a la docencia con la finalidad de propiciar el incremento en la efectividad del proceso de enseñanza-aprendizaje de la asignatura de Matemáticas I. Además de observar como función deseable no servir solamente a nuestra Facultad sino también a aquellas Facultades o Escuelas que hacen contemplativa la enseñanza de la lógica matemática.

Se propone emplear a la computadora como un medio que proporcione instrucción y presente oportunamente lecciones de ejercicio-práctica las cuales son adecuadas para incrementar los logros académicos cuando es utilizada como una herramienta de apoyo y no así como un sustituto en las aulas ni mucho menos como suplente del docente.

Los alumnos logran aprender más y a un nivel más rápido con la ayuda de la Instrucción Asistida por Computadora auxiliados por el profesor como una alternativa adicional aparte de la enseñanza acostumbrada que les es impartida en el aula.

El empleo de sistemas tutoriales también conduce a actitudes más positivas y aumenta el interés por otras aplicaciones en la computación.

Como objetivo terminal de este proyecto de tesis se sugiere la elaboración de un sistema tutorial que le permita al alumno tener una experiencia agradable para el aprendizaje de los temas de lógica matemática que estructuran la asignatura de Matemáticas I, dándole la oportunidad de profundizar más en la Lógica Matemática y auxiliando al docente en su tarea formativa de adiestramiento y control.

El sistema tutorial se elaboró para que le permitiera al alumno contar con un apoyo y una acción interactiva en su aprendizaje; haciéndose para ello necesaria una investigación previa de los diversos medios que ofrece la computadora, así como del software específico, además de consultar a profesores y emplear bibliografía que conllevará a la elaboración de la parte explicativa del tutorial.

En el primer capítulo de este trabajo se presenta el conocimiento generado en torno a los antecedentes de la Instrucción Asistida por Computadora (IAC).

El capítulo dos describe los medios y recursos disponibles dentro del desarrollo de software educativo.

---

1 Plan de Estudio 1993 (Tomo I), México, UNAM, FCA, Mc Graw-Hill, 1993, pp. 7-69.

Los requerimientos preliminares originados a partir de la problemática se definen en el capítulo tres. La propuesta para elaboración de un prototipo que concibe paralelamente los requerimientos finales del diseño del sistema propuesto, el desarrollo de la metodología aplicada dentro de la creación del sistema, los diagramas que simbolizan el flujo de datos de los procesos, el contenido de la información de cada proceso, la estructura de los módulos que integran al sistema, los diagramas de proceso de cada módulo así como la estructura de la base de datos y su interrelación con los subsistemas que conforman al sistema tutorial; también están contenidos en el capítulo.

En el capítulo cuatro se mencionan los aspectos generales del lenguaje de programación utilizado para la codificación, el estilo y los estándares aplicados a los programas así como el equipo utilizado tanto en el desarrollo como en la implementación del sistema tutorial.

Los resultados del desarrollo y la evaluación del sistema tutorial son presentados en el quinto capítulo.

En el sexto capítulo se presentan las conclusiones derivadas del presente trabajo y son sugeridas algunas propuestas sobre desarrollos a ser utilizados dentro del ámbito educacional valiéndose de la computadora como apoyo educativo.

Finalmente se anexan un ejemplo de código fuente, la ficha de evaluación empleada durante la validación del sistema tutorial y el manual de usuario del mismo.

Es así como esta tesis ha sido preparada para cumplir el propósito de servir como libro de consulta para realizar sistemas tutoriales y como apoyo en la asignatura de Matemáticas I.

Deseo que este trabajo motive a grupos posteriores de Seminarios de Investigación en Informática a la investigación y al estudio que los encauce a penetrar más en el desarrollo de programas educativos, ya que es un tema tan basto que difícilmente podría ser agotado por un solo trabajo de tesis.

I. INSTRUCCION ASISTIDA POR COMPUTADORA.

1.1 Antecedentes de la Instrucción Asistida por Computadora.

Desde los años 50 el software educativo consistía en paquetes de información, preguntas al respecto e instrucciones para el cambio de secuencia de acuerdo con respuestas dadas. En los últimos años se ha considerado a la computadora como apoyo instruccional dando al software un carácter de autocontenido sustentado en un enfoque educativo, en el cual, la computadora presenta la información e interactua con el estudiante para mejorar el aprendizaje. Actualmente los programas computacionales en esta modalidad, orientados hacia el contenido educativo cumplen los objetivos de estudio y están elaborados de acuerdo con el curriculum, su función es activar el proceso de enseñanza-aprendizaje y cuentan con modalidades que se denominan formas IAC.

Los principios de la Instrucción Asistida por Computadora se derivan del uso de la computadora como apoyo instruccional. En los Estados Unidos se realizaron los primeros esfuerzos para aplicar las computadoras en la educación; observándose una evolución continua en cuanto a la concepción que se tiene de su empleo. Esta se vio fortalecida por el avance tecnológico de los equipos y las exigencias que se tenían por parte de los usuarios en las instituciones de educación.

El matemático y filósofo Patrick Suppes de la Universidad de Stanford establece que la verdadera función revolucionaria de las computadoras en la educación se debe a la Instrucción Asistida por Computadora. [1]

En los orígenes de la IAC están las ideas conductistas de B.F. Skinner [2] sobre el aprendizaje. Skinner estudia el aprendizaje animal en condiciones de laboratorio. Por ejemplo, el estudio de la conducta de un animal hambriento que al oprimir una palanca obtiene comida, la conducta de oprimir la palanca sea cada vez más probable, estos mismos principios pueden aplicarse al aprendizaje humano. Si el alumno proporciona una respuesta y esta es reforzada diciéndole que es correcta con una musiquilla, o bien con una imagen animada la probabilidad de aprender aumentará. Skinner defendió la enseñanza programada y la utilización de las computadoras para apoyo instruccional como una vía para salir de la crisis educativa. La postura de Skinner encuadrada en la corriente denominada "tecnología educativa" ha sido y es un tema controvertido.

Sin embargo, existen otras teorías como la "teoría cognitiva" que, con argumentos muy distintos validan las potencialidades educacionales de las computadoras dentro de la enseñanza.

Gran parte de las tecnologías y medios educativos, como lo es el utilizar a las computadoras como herramientas intelectuales han tenido sus raíces en la psicología cognitiva. Pero tener una idea de como funciona la mente no significa tener una idea de aprendizaje adecuada. Las teorías cognitivas son prometedoras pero por ahora no siempre resultan acertadas para desarrollar los módulos de aprendizaje.

José M. Arias [3] señala que hubo un instante en el que parecía que las computadoras iban a traer la solución a todos los problemas de la enseñanza, además de poder dar una enseñanza individualizada e incluso que iban a sustituir al profesor, de ahí un cierto rechazo de algunos profesores a las computadoras en su momento.

---

1 En Arias, M. José. "EAO (Enseñanza Asistida por Ordenador)", Nuevas Tecnologías, España, Anaya, 1981, p. 2.

2 Ibid. p. 2.

3 Ibid. p. 2.

E. Carbajal, y C. Pesina [4] establecen que en México la incorporación de las computadoras a la educación requirió de una serie de cambios que adecuaron de los nuevos conocimientos al entorno social. Aludiendo al futuro de nuestro sistema educativo y a la mencionada "modernidad" dentro del contexto de la actual política educativa es importante reflexionar acerca de la filosofía que sustenta el uso de la computadora como herramienta didáctica, como objeto de estudio y la relación entre sujeto, objeto y conocimiento.

1.2 El empleo de la computadora en la adquisición o dominio del conocimiento.

El empleo de la computadora en la educación tiene dos fines: [5]

1. Facilitar la adquisición del conocimiento.
2. Proporcionar el dominio del conocimiento.

Para establecer las diferentes formas de utilizar a la computadora como un valioso auxiliar en la educación, es necesario primero plantear las diferencias entre lo que es la adquisición del conocimiento y lo que se puede denominar como el dominio del conocimiento.

Las diferencias entre adquisición y dominio del conocimiento se pueden establecer en cuatro planos distintos: por sus objetivos, por su contenido, por ordenamiento y por la forma en que se hace su evaluación.

Objetivos y contenido.

Cuando se trata de adquisición del conocimiento se refieren a su acumulación partiendo del supuesto de que dichos conocimientos son útiles para el desarrollo de una determinada actividad futura. Por lo general, se hace referencia a un temario de amplio contenido y con una estructuración lógica, la cual debe impartirse, en un determinado tiempo después del cual el alumno debe haberlo memorizado o comprendido.

Si se trata de dominio del conocimiento, se esta haciendo referencia directa e inmediata sobre la capacidad de manejar conocimientos para la resolución de problemas o para su aplicación. El contenido se encuentra constituido por una selección de hechos, reglas y relaciones, así como un conjunto de procedimientos que aseguran la realización de determinadas tareas o resolución de problemas.

En algunas áreas del saber es muy clara la relación y mezcla de estos dos tipos de conocimientos, como lo es, por ejemplo, durante la enseñanza de la física o de la termodinámica, asignaturas en las cuales la explicación de conceptos y relaciones va seguida de ejemplos donde se emplean dichos conceptos, después de los cuales se deja como tarea la realización de problemas ad hoc para que el alumno demuestre que no sólo adquirió los conocimientos sino que los domina lo suficiente para su empleo en la resolución de problemas. En resumen, el contenido de programas para que el alumno adquiera conocimientos, se encuentra constituido por temarios relativos a un campo del saber con una determinada estructura y sin relación directa con su empleo.

Por otro lado un programa orientado para proporcionar dominio del conocimiento, se puede considerar constituido por los conocimientos relevantes y seleccionados que aseguran la resolución de un determinado tipo de problemas, más un conjunto de ejercicios con cuya práctica se asegura que el objetivo fue alcanzado.

---

4 Carbajal, E. y Pesina, C. "La Computación en la Enseñanza", Ciencia y Desarrollo, México, CONACyT, No. 88, Sep.-Oct. 1989, pp. 124-134.

5 Gutiérrez, Alberto. "La Computadora en la Enseñanza de Métodos Generales de Razonamiento : El Sistema Minerva" Influencia de los Sistemas Modernos de Información en el Desarrollo Educativo (Memoria 4a. Conferencia Internacional), México, UNAM, 1988, pp. 137-138.

La computadora y la forma de evaluación.

En cuanto a la forma de evaluar los objetivos educacionales en los programas destinados a proporcionar adquisición de conocimientos, esta evaluación se hace por medio de preguntas que aseguran que el alumno ha adquirido dichos conocimientos.

Cuando se trata de evaluar el dominio del conocimiento, el alumno debe demostrar su capacidad de emplear dichos conocimientos en la resolución de problemas que requieran su empleo.

Gutiérrez finalmente señala, si la computadora se utiliza para asegurar la adquisición de conocimientos, se esta refiriendo a programas en los cuales se hace una presentación interactiva en la pantalla de los diferentes conceptos que constituyen el contenido de la materia con preguntas intercaladas que aseguran que va comprendiendo la información previamente presentada, como es el caso del empleo de material estructurado bajo los lineamientos de la enseñanza programada. En cambio, si se emplea la computadora para asegurar el dominio de un conocimiento, se esta haciendo referencia a un sistema capaz de generar un problema relativo al campo del saber del que se habla, y que permite probar al usuario si sus conocimientos le son suficientes para resolver dicho problema, al mismo tiempo que el programa tiene la capacidad de la resolución de dicho problema, como ocurre con los sistemas tutoriales inteligentes.

1.3 La computadora y las teorías del aprendizaje.

Uno de los aspectos más importantes en las presentaciones que se hacen sobre computadoras y educación es el de las teorías de aprendizaje que sirven como soporte a la elaboración y utilización del software. No es finalidad de este trabajo hacer una presentación exhaustiva y profunda del tema, pero, es substancial que cualquier proyecto de informática educativa presupone una filosofía de base que no debe olvidarse en ningún momento del desarrollo o evaluación de un sistema educativo. [6]

Dentro de la psicología experimental han surgido en este siglo dos corrientes que fundamentan los proyectos de aplicaciones de la computación en la educación; el neoconductismo y el cognoscitivismo.

Para el neoconductismo el motor del proceso de aprendizaje es el refuerzo. Este se presenta cuando la conducta de una persona es recompensada de alguna manera: (el refuerzo se otorga cuando se presenta la conducta); el aprendizaje se da cuando se modifica la conducta que como requisito debe ser observable.

Bajo el concepto de esta teoría un individuo aprende, es decir modifica su conducta, observando las consecuencias de sus actos, desde este punto de vista las consecuencias que aumentan la probabilidad de una conducta son refuerzos. Una conducta se fortalece en relación a la frecuencia del refuerzo, por lo que el comportamiento de un alumno puede ser estructurado por un reforzamiento diferencial (donde se fortalecen las conductas deseables y no se fortalecen aquellas a las que se quieren evitar) que desglose las conductas complejas en actos simples.

El cognoscitivismo, por su parte, reconoce el aprendizaje como un proceso, es decir, lo observa como una dinámica de estructuración del conocimiento (aclarando en términos de precisión, que existen diferentes modelos en el cognoscitivismo).

En esta teoría se considera que el individuo incorpora el nuevo conocimiento a su propia estructura cognoscitiva (modelo perceptivo de su experiencia pasada que está como referencia a la actual), y por lo tanto el aprendizaje se da no sólo al responder al estímulo sino desde el momento en que se tiene contacto con este; el individuo aprende cuando incorpora la nueva información. Bajo el concepto de esta teoría un

---

6 Laguna, Rubén. "La Computadora como Apoyo Didáctico", Influencia de los Sistemas Modernos de Información en el Desarrollo Educativo (Memoria 4a. Conferencia Internacional), México, UNAM, 1988, pp. 165-167.

individuo aprende cuando encuentra en el nuevo conocimiento un propósito en su experiencia y necesidades; cuando se le presenta la información de tal manera que le permita reorganizar su estructura cognitiva y cuando se le aclara cómo y porqué se han producido sus errores y no sólo se le refuerza.

Adicionalmente a estas dos propuestas Rubén Laguna (1988) trata de manera separada la teoría genética de Piaget (ubicada también dentro de la escuela cognoscitivista) que se refiere a los mecanismos que involucran la adquisición del conocimiento, en función del desarrollo del individuo; en este sentido la teoría se enfoca al estudio del nivel de adquisición del conocimiento, es decir, como pasar de un estado de "menor conocimiento a uno de mayor".

Su idea central es que el desarrollo intelectual constituye un proceso adaptativo de "asimilación" y "acomodación" que se distingue por estados caracterizados por una estructura matemática de conjunto.

Tomando en cuenta estas tres posiciones se puede identificar bajo que teoría se encuentra un software educativo y su aplicación a partir de los siguientes principios [7] (los tres primeros pertenecen al neoconductismo, los tres segundos al cognoscitivismo y el último a la teoría genética):

1. Principio de actividad.

   El alumno debe hacer aquello que se espera que aprenda, es decir, conseguir él mismo su aprendizaje.

2. Principio de progresión de la dificultad.

   El avance o progresión debe estar en función del progreso del alumno; el ritmo particular del estudiante ha de respetarse.

3. Principio de verificación inmediata o seguridad de aprendizaje.

   El alumno debe enterarse de los aciertos o errores que tenga su aprendizaje; esto debe ocurrir lo más pronto posible.

4. Principio de actividad propositiva.

   El alumno aprende mejor las cosas que realiza cuando tienen un significado para él.

5. Principio de organización por configuraciones globales.

   Organiza el material que será presentado al estudiante para que se "apropie" de la información incorporándola a su estructura cognoscitiva a fin de favorecer el aprendizaje.

6. Principio de retroalimentación.

   No es tan importante el número de aciertos o errores que haya cometido el alumno como el que se aclare cómo y porqué se han producido.

7. Principio de adecuación al estudiante.

   El material debe presentarse al alumno considerando su nivel de desarrollo cognoscitivo.

---

7 Ibid. p. 166.

2. MEDIOS Y RECURSOS PARA EL DESARROLLO DE PROGRAMAS EDUCATIVOS.

2.1 Lenguajes de programación, lenguajes especiales, sistemas y programas de autor.

El desarrollar eficientes programas didácticos no es tarea fácil. El uso de un lenguaje o sistema de autor puede ser una experiencia accesible tras pocas sesiones de práctica. Pero dado el nivel de complejidad que poco a poco va alcanzando la creación de programas, es preciso conocer las características de los lenguajes o sistemas que se adoptan para saber cual satisface los requerimientos docentes en casos concretos. [1]

De acuerdo al orden de facilidad creciente, los recursos para la producción de software educativo son:

- Lenguajes de programación.
- Lenguajes de autor.
  - Para computadora central y terminales.
  - Para microcomputadoras.
- Sistemas de autor.
- Programas de autor.

Cualquiera de estos recursos puede estar vinculado a otros en la tarea de crear un programa.

Las figuras 2.1 a 2.4 que se presentan a continuación esquematizan las tablas descriptivas de los recursos para la producción de software.

---

1 Giordanio, E. y Edelstein, R. La Creación de Programas Didácticos, España, Gustavo Gili, 1987, pp. 59-109.

Figura 2.1 Lenguajes de programación

Lenguajes de programación.	Basic. Código simbólico de Instrucciones de Aplicación General para Principiantes creado en Darmouth, en los años 60 por Kemeny, J. y Kurtz, T. como un lenguaje para la enseñanza. Es un lenguaje interactivo cuya gran popularidad se debe a que es un lenguaje fácil de usar. Fue desarrollado inicialmente para alumnos y poco a poco su campo de acción se extendió al uso de aplicaciones administrativas e industriales. Actualmente existen versiones más sofisticadas que la original, ejemplos de ellas son as adaptaciones en la mayoría de las microcomputadoras personales de bajo costo.
Representación de símbolos , caracteres y reglas de uso que permiten a las personas comunicarse con las computadoras. Algunos lenguajes son creados para una aplicación especial mientras que otras son herramientas de uso general más flexibles que son apropiadas para muchos tipos de aplicaciones. Los lenguajes de programación tienen instrucciones ya familiares de entrada/salida, cálculo/manipulación de textos, lógica, almacenamiento y recuperación.	C. Desarrollado por Richie, D. a principios de los años 70's. Es un lenguaje de alto nivel de aplicación general similar al ensamblador que provee de bibliotecas y funciones que facilitan el manejo de ambientes gráficos y base de datos. Los programas escritos en este lenguaje pueden transferirse de una arquitectura de máquina a otra sin mayor problema , los programas elaborados en C pueden emplear conceptos modulares y estructurados.
	Pascal. Desarrollado por Wirth, N. a finales de la década de los 60's. Fue creado a partir de Algol y denominado en honor de Blas Pascal. Diseñado para la enseñanza de la programación como una disciplina sistemática, es el primer lenguaje de programación importante que se creó en base a los conceptos de programación estructurada. Se convirtió en un lenguaje estándar en 1983. Actualmente se han desarrollado versiones más poderosas que la original, la mayoría de las cuales trabajan en microcomputadoras comerciales por lo que con el paso del tiempo se ha convertido en un lenguaje muy popular.

Figura 2.2 Lenguajes de autor

Lenguajes de autor. Software que posibilita la creación de material didáctico con un número limitado de instrucciones de programación, reduciendo al mínimo la preocupación del profesor por la técnica informática. Un lenguaje de autor en cuanto al nivel de complejidad de sus posibilidades combinatorias con frecuencia no difiere mucho de un lenguaje de programación.	Para computadora central y terminales. Generalmente enmarcados en lo que suele llamarse un entorno o sistema informatizado de enseñanza se han venido utilizando desde los años 70 en diversas universidades principalmente norteamericanas. La mayoría de las estos lenguajes y entornos se inclinan hacia la línea de producción de un courseware. Una de las ventajas que en general reconoce al uso de tales entornos es la abundancia de programas didácticos para el aprendizaje de alguna asignatura. Es importante destacar que los programas creados por el profesor y las respuestas del alumno se almacenan necesariamente en la computadora central que por fuerza escapan al dominio exclusivo de cualquier individuo. Para microcomputadora. Existen varios lenguajes de autor realizados especialmente para microcomputadoras o adaptados para este uso. La programación con un lenguaje de autor varía según el tipo de lenguaje y sus características. En algunos casos, programar con uno de estos lenguajes es casi como hacerlo en una sencilla variante del lenguaje Basic. Otras veces, la programación se simplifica considerablemente aunque al costo de una mayor rigidez en los formatos de las pantallas y actividades.

Figura 2.2.1 Lenguajes de autor para computadora central y terminales

TUTOR (entorno PLATO, Programmed Logic for Automatic Teaching Operation, o Programmed Learning and Teaching Operations).
Tal vez el más evolucionado de cuantos se han desarrollado hasta ahora para la IAC. PLATO es el nombre de un entorno informático destinado hasta ahora fundamentalmente a aplicaciones educativas de nivel universitario. Desarrollado a finales de los años 60 y principios de los 70 por Control Data Corporation. El lenguaje de autor TUTOR se acabó de desarrollar en 1967, como núcleo central del proyecto PLATO, que utiliza un sistema informático como fuente alimentadora de todo el entorno educativo. Desde el punto de vista tecnológico, el entorno PLATO ha incorporado los mayores avances del desarrollo informático ya que incorpora el uso de la visualización de imágenes en color mediante un proyector de acceso directo y las pantallas son sensibles la tacto del alumno, que puede responder tocando un punto u otro de su superficie. Otro de los rasgos que caracterizan a este entorno es la posibilidad de registrar, por ejemplo durante todo un semestre, la puntuación y las respuestas de los alumnos en una asignatura.

APT y TAL (entorno TICCIT).
El proyecto TICCIT (información televisada controlada por ordenador de modo interactivo y a tiempo compartido) es otro sistema de gran envergadura, destinado como PLATO a la utilización de programas de IAC con apoyo visual. Proyecto financiado por la National Science Foundation de los Estados Unidos, este presupuesto ha incluido el desarrollo de software, que ha pasado a ser de dominio público y que ahora se suministra gratuitamente a las escuelas de EUA. TICCIT considera separadamente los tres niveles de trabajo que intervienen en el diseño y la creación de programas. El objetivo es reservar al docente la elaboración de contenidos didácticos, mientras que la programación y el modelo de enseñanza vienen dados en gran medida por el sistema. El modelo de enseñanza asociado al sistema TICCIT suele describirse como estrategia de exposición de contenidos-ejemplificación-secuencias de práctica y a grandes rasgos puede considerarse subsidiario de la enseñanza programada del llamado aprendizaje conceptual. Los lenguajes de autor APT y TAL, disponibles para el entorno TICCIT (computadora Date General), no son aplicables a ninguna otra clase de computadoras.

CAN 8.
Desarrollado por le firma CLL-Honeywell para sistemas informáticos centralizados. Su lección escrita se define como conjunto de enunciados. Las instrucciones que materializan dichos enunciados pertenecen a las categorías:
- Instrucciones de presentación del material didáctico (presentación de textos y gráficos, así como la posibilidad de añadir comentarios a los programas creados).
- Instrucciones de interacción con el estudiante (incluyen las de entrada y análisis de las respuestas, las de control de resultados y las instrucciones de encadenamiento del programa). CAN 8 es un buen precursor de los lenguajes de IAC, dado que la sintaxis es sencilla y el número de instrucciones bastante limitado.

LOVE.
Es una de las primeras experiencias europeas en el campo de lenguajes informáticos pensados para educadores. Desarrollado por investigadores de la Universidad de Lyon sólo incluye un total de doce instrucciones, lo que facilita enormemente su uso. Estas instrucciones se escriben en lenguaje casi natural, representadas con las primeras letras de las palabras francesas que las definen. El lenguaje solamente funciona en sistemas informáticos centralizados y no esta por ahora disponible en países de lengua castellana. Dada la evolución actual de la informática educativa hacia el uso de microcomputadoras probablemente el lenguaje LOVE nunca llegue a desarrollarse en castellano en su versión actual.

Figura 2.2.2 Lenguajes de autor para microcomputadoras

PEPA MACA (Preparador, Editor y Procesador de Autoevaluaciones de la Máquina para el control de aprendizaje).
Este lenguaje está escrito en Fortran IV y tiene su origen en un centro universitario ya que este fue desarrollado por profesores del Centro de Cálculo de la Universidad politécnica de Barcelona, a fin de disponer de una herramienta útil para llevar a cabo distintos proyectos de IAC. Su área de aplicación se circunscribe principalmente a la creación de bases de datos de "elementos de interacción tutorial con el alumno". PEPA MACA se define como un sistema que "consiste en la confección de cuestionarios interactivos de Instrucción Asistida por Computadora", basados en un sencillo esquema de diálogo. Las reglas de sintaxis son muy pocas y la misión del lenguaje "es permitir al profesor la redacción de material didáctico concentrándose en los aspectos educativos". Una ventaja de este lenguaje de programación, es la existencia de "compiladores cruzados" (cross compilers) que permiten la traducción del código fuente para su ejecución bajo el control de ordenadores de distinto tipo.

PILOT (Programmed Inquiry, Learning and Teaching).
Es uno de los lenguajes de autor más conocidos y existe desde ya hace varios años en su versión adaptada para diversas microcomputadoras (Apple II, PC compatibles, etcétera). La versión original data de 1968 y fue desarrollada en la Universidad de California por J. Strawether y su equipo de colaboradores. Inicialmente este lenguaje sólo disponía de ocho instrucciones, lo que facilitaba enormemente sus uso. Una versión más actual más conocida (PILOT y SUPER-PILOT para la computadora Apple), dispone de 32 instrucciones y es mucho más compleja y potente. Además, es posible extender sus posibilidades incorporando características de gráfico y sonido. PILOT se define como un lenguaje pensado para crear programas de IAC individualizados e interactivos. En su versión más conocida PILOT y SUPER-PILOT disponen de 32 instrucciones y es mucho más compleja y potente. Sus posibilidades pueden extenderse incorporando nuevas prestaciones de gráficos y sonido. Sus instrucciones y tienen esencialmente dos partes: la instrucción en sí misma y el objeto de la instrucción. El lenguaje contempla el uso de modificadores que influyen en la ejecución de una instrucción y de cláusulas condicionales que determinan si la instrucción debe o no ejecutarse en diferentes condiciones, lo que permite incluir numerosas ramificaciones para las distintas respuestas del alumno.

LAC-DOS.
Se define también como "un programa diseñado para la creación, aprendizaje y evolución de cursos de enseñanza". Desarrollado en España por CIESA, distribuidora de computadoras Olivetti y software para la enseñanza. Los componentes de su entorno son un programa generador de textos y actividades, un sistema de creación de gráficos, una utilidad para organizar o modificar las lecciones, un programa de presentación de las lecciones para los alumnos y un tratamiento de los ficheros de los resultados desarrollados por los alumnos. Es apto para la creación de aquellos programas que requieren mezclar el texto con gráficos y que emplean algunas preguntas de respuesta breve que no requieren análisis. Una de sus ventajas es la admisión de múltiples entradas del alumno en la pantalla.

ARLEQUIN.
Es llamado sistema de autor y comprende cerca de treinta instrucciones. Uno de los aspectos que en el más destacan es sus estructura de datos. Tiene la particularidad de definir "listas" para usar un esquema de evaluación. Esta característica conjuntamente con el complemento habitual de comodines y delimitadores le da un potencial esquema de evaluación muy preciso y rico en la aceptación de los diversos matices posibles en la respuesta del alumno.

MICROTEXT.
Desarrollado por la National Physical Lybrary de Inglaterra, por la computadora BBC entre otras. Sus instrucciones se componen de subconjuntos de las sentencias EDUTEX, lenguaje de autor desarrollado para computadoras más grandes. Su organización se basa en cuadros o pantallas de información numerados. Tiene rutinas para la evaluación de respuestas y utiliza la bifurcación del programa como un elemento de interacción con la respuesta del alumno.

Figura 2.3 Sistemas de autor

Sistemas de autor. Es un conjunto de programas compatibles entre sí, que permiten crear programas didácticos sin saber informática y se caracteriza por una mayor sencillez en las instrucciones. Estos se presentan en forma de preguntas u opciones en lugar de escribirse en forma de código y por un nivel de interacción más próximo al usuario de programas que al programados. En este sentido los sistemas de autor guardan alguna semejanza con los lenguajes de autor, por ser más flexibles que los programas de uso específico o de formato estándar. Su diseño se apoya en un modelo de enseñanza que presenta la información en forma de texto, con la subsecuente comprobación de la compresión del alumno. En otros casos el texto es el soporte de una serie de operaciones que se desarrollan sucesivamente en una o más pantallas y constituye sólo el contexto de la actividad. A menudo, estos programas sólo admiten respuestas cerradas, en forma de respuesta exacta: cuestionarios de elección múltiple, actividades de asociación o emparejamiento y opción verdadero-falso; o bien permiten completar cierto número de caracteres (letras, palabras, frases o signos) dejados en blanco. No, obstante algunos sistemas admiten varios esquemas de evaluación de la respuesta, lo que permite una interacción más amplia al posibilitar el análisis de respuestas más libres de los alumnos. El uso de comodines (signos que permiten remplazar cualquier letra) sirve para ampliar el repertorio de respuestas previstas y da a algunos sistemas una mayor tolerancia ortografica. Los sistemas de autor pueden contar con recursos complementarios, tales como pantallas de ayuda, glosarios facilmente programables y otras utilidades que mejoran la presentación del material didáctico (por ejemplo, para desarrollar gráficos y efectos sonoros). La posibilidad que brindan algunos sistemas de conservar los resultados del alumno seria muy difícil de programar con lenguajes de autor, a excepción de PILOT.	PRIVATE TUTOR. Desarrollado por T. Brown, R. Cabell, M. Kechula, J. Shear, se ha utilizado para la realización de programas tutoriales en el aprendizaje del lenguaje BASIC y del uso de la computadora IBM. También se presta a la creación de programas de IAC para cualquier asignatura. El conjunto del sistema , que no se ha publicado en versión castellana consiste en cuatro programas de diferente función. (Presenter, Preparer, Reporter y Author). Este sistema está pensado para trabajar dentro de límites que no son aceptados para la creación de programas educativos, ya que se apoya fuertemente en el concepto de diálogo como la principal forma de interacción didáctica. No obstante, se considera que la clave de su utilidad estaría en la posibilidad de que el usuario se sirviese de aquellas características que le dan una cierta medida de flexibilidad al sistema, aprovechando al máximo lo que este sistema pueda dar de si dentro de sus limitaciones. Queda margen para realizar algunos programas acordes con la metodologías preferidas por los autores y con sus objetivos pedagógicos. EDULOGIC. Desarrollado por la editora canadiense IDEATECH, de Quebec y programado en lenguaje ensamblador 8088. Es un sistema de uso general diseñado para microcomputadoras PC compatibles. Los requerimientos de este sistema de autor por el momento son bastante costosos y no están al alcance de cualquier centro docente. Dada su reciente aparición, aún no se conocen resultados concretos plasmados en forma de programas de aplicación didáctica. Sin embargo, este sistema de autor parece bastante flexible y ofrece un gran número de utilidades para la creación de programas. Aunque se define como "sistema-autor", reúne la mayoría de las características propias de los lenguajes de autor. Esto puede resultar tanto ventajoso en ciertas circunstancias como desfavorable para los educadores no iniciados en la programación de IAC. Dada la complejidad potencial de su uso y la utilización de comandos que remiten a manuales o múltiples pantallas de explicación. VISA. Ha sido desarrollado por Langage et Informatique, una editora de la ciudad de Toulose, Francia. Está escrito en lenguaje LSE, lo que limita su uso a computadoras marca Thompson MO5 y TO7/70 (y para Nanoréseaux), que no son compatibles con otras microcomputadoras y cuyo uso no es muy frecuente fuera de Francia. Una potencial prestación que ofrece el sistema es, la de controlar un proyector de diapositivas o un sistema de videodisco. En el plano de sus limitaciones, además de los que le impone el lenguaje LSE, se le reprocha en Francia la lentitud con que trabaja el editor de textos y la sumariedad de la documentación que acompaña a este sistema de autor.

Figura 2.4 Programas de autor

Programas de autor. Diseñados específicamente para la creación de ejercicios y actividades en base a uno o más formatos fijos. Estos programas tienen una flexibilidad limitada en cuanto a opciones de diseño y en general son programas pensados en primer término para el aprendizaje de idiomas, no obstante, muchos de estos programas tienen una aplicación de tipo general ya que los ejercicios que proponen son clásicos y bien conocidos en el marco de cualquier asignatura.	QUESTIONMASTER (Cuestionario y Programa de Evaluación). Desarrollado por la editorial británica Wida Software. Pese a sus obvias limitaciones, ilustra bien la posibilidad de utilizar programas de autor diseñados para la enseñanza de lenguas en otras asignaturas. Se presenta como adecuado para tratar cualquier tema basado en un texto. Permite crear sus programas tecleando un texto de introducción (de hasta 50) y a continuación un cuestionario de 40 o menos preguntas. MATCHMASTER (Actividades de Asociación o Emparejamiento). Es un ejemplo de programa de diseño estándar para uso tan específico como es el de realizar comparaciones de dos clases de ítems, a semejanza de los clásicos ejercicios de asociación o emparejamiento. Fue realizado por C. Jones, autor de programas para la enseñanza de idiomas. La validez de este programa no se limita forzosamente a esa asignatura, ya que este modelo de actividad puede ser útil a cualquier área de conocimiento. CHOICEMASTER (Cuestionarios de elección múltiple). Diseñado para la creación de cuestionarios de elección múltiple. Es fácil de usar. Intenta ser una herramienta útil dentro de las limitaciones de su formato fijo e incluye suficientes características que van más allá de los típicos cuestionarios de elección múltiple por escrito. Sus comentarios para las respuestas incorrectas son de evidente utilidad para crear programas tutoriales de autor auto-correctivos o de recuperación.

2.1.1 Comparaciones entre los recursos para producción de software educativo.

E. Giordanio y R. Edelstein [2] señalan que cada uno de los lenguajes de programación o de autor así como los sistemas y programas de autor comentados anteriormente, tienen características propias que los diferencian de los restantes. Algunos lenguajes de autor por su relativa sencillez no son aptos más que para usos muy concretos, como la creación de cuestionarios con el lenguaje de autor PEPA MACA. Pero esos usos pueden extenderse creativamente hacia diversas asignaturas, como se ha demostrado algunas personas que utilizando este lenguaje trabajaron en el área de la enseñanza del español, informática, etcétera. Esto pone en duda la utilidad de un lenguaje de autor como PEPA MACA. Solo que no parece muy deseable que para cierto tipo de asignaturas que los futuros alumnos recurran a un plan basado en una infinita serie de preguntas de opción múltiple, aunque estén apoyadas por la proyección individual de diapositivas.

El sistema PRIVATE TUTOR y el lenguaje PILOT incluyen esquemas para reconocer distintas variantes ortográficas de las respuestas del alumno. El esquema general de PEPA MACA admite esta posibilidad, pero puede costar más trabajo el prever y reconocer las variantes. Una limitación muy importante de casi todos los lenguajes de autor, que repercute en la enseñanza de idiomas, es la imposibilidad de trabajar en un contexto capaz de reacomodarse tras cada intervención de alumno. Esto elimina la posibilidad de realizar ejercicios con múltiples blancos, actividades de reconstrucción de textos, juegos de ordenamiento de las partes de un texto en el marco de la pantalla y otras actividades son valor lúdico, como la corrección de errores ortográficos, sintácticos o de puntuación en un contexto motivador.

En el campo de la realización de gráficos, los lenguajes de autor como PILOT y LAC-DOS y los sistemas de autor como EDULOGIC tienen sus ventajas y este es un buen argumento para su uso. El lenguaje de autor PEPA MACA no admite gráficos ni ofrece posibilidad alguna de utilizar sonido.

No obstante, hay cierto nivel de complejidad instrumental para la creación de gráficos con cualquiera de estos lenguajes y sistemas. Lo mismo ocurre con las prestaciones para crear efectos sonoros. Crear una melodía en PILOT implica conocer el código de cada nota en el teclado y combinarlo adecuadamente en el programa. Esto no es demasiado difícil pero tampoco resulta fácil.

El desarrollo de programas más sofisticados no garantiza mejores aplicaciones de la IAC si los autores no se esfuerzan por obtener los beneficios que los sistemas actuales podrían dar de sí. Este es ante todo un problema de metodología. Es preciso explorar nuevos caminos metodológicos para comprobar la validez de la computadora en el proceso de aprendizaje. Una posibilidad atractiva sería la de acompañar a cada sistema o lenguajes de manuales de metodología aplicada a la IAC e ideas de tipología de ejercicios aunque esto puede ser impracticable en más de una asignatura.

2.2 Programas educativos.

A la mayoría de los programas de aplicación didáctica se les designa generalmente como programas de IAC, exceptuando el aprendizaje de un lenguaje de programación y excluyendo también las aplicaciones propiamente informáticas de uso educativo (bases de datos, procesadores de texto, paquetes de contabilidad, etcétera.)

Al margen de los ambiciosos paquetes de software de enseñanza programada, tanto lineal como ramificada, el concepto de IAC suele abarcar un variado repertorio de aplicaciones educativas, muchas de ellas potencialmente interesantes.

---

2 Ibid. pp. 63.

A continuación se describen lo que E. Giordanio y R. Edelstein [3] consideran algunos de los usos más corrientes de cada una de esas aplicaciones, sin pretender una clasificación exhaustiva de las muy variadas formas que hoy adopta la IAC.

2.2.1 Ejercicios y prácticas por computadora.

Existe un campo de aplicación muy amplio para las prácticas y ejercicios realizados por computadora, siempre y cuando estos programas de IAC se adapten flexiblemente a la programación y metodología de cada docente, en lugar de imponerle a este una opción metodológica invariable. Muchos autores coinciden en señalar que la computadora liberará al profesor de las tareas más repetitivas, monótonas y rutinarias de la enseñanza. El factor motivación del estudiante es fundamental para juzgar el valor de estos programas.

En las áreas de lengua y matemáticas, la ejercitación de ciertas habilidades es siempre imprescindible. Las modernas teorías de aprendizaje han limitado el uso de las baterías de drills, y de los antiguos métodos de refuerzo estructurado, inclinándose por metodologías que ponen al estudiante en situación de sujeto de aprendizaje y no de receptor pasivo. Pero el uso de la computadora y el florecimiento de lenguajes y sistemas de autor, conlleva al peligro de volver a esos métodos, por la extrema facilidad con que se pueden crear programas de esta clase. Sin embargo, la computadora tiene planteado un desafío en la renovación de los métodos de enseñanza y de pruebas para poder superarlo.

Los ejercicios de asociación y emparejamiento, por ejemplo, pueden ser entretenidos y eficaces como forma de trabajo. Los distintos ítems encolumnados desordenadamente (para que los alumnos los agrupen por parejas presionando una tecla), pueden consistir en palabras con sus correspondientes sinónimos y antónimos, regiones y climas, operaciones y resultados, o cualquier otra materia que permita establecer relaciones entre pares de elementos. Este formato de ejercicio también sirve para realizar actividades de descubrimiento, como forma de presentación que permite al alumno descubrir información o establecer nuevas relaciones.

Corresponderá siempre al profesor determinar si el ejercicio justifica el uso de la computadora, pizarrón o si, por el contrario su realización sería más práctica utilizando medios convencionales (papel y lápiz, pizarrón, diálogos de viva voz, acetatos, etcétera.). Para tomar esta decisión, el profesor deberá tener en cuenta las principales ventajas de un sistema informatizado que, pueden ser aplicables o no, en función de cada objeto de aprendizaje.

2.2.2 Programas de simulación.

Los programas de simulación pueden convertir a la computadora en un microlaboratorio artificial. A diferencia de los programas de demostración basados en un tipo de exposición muy poco interactiva, las simulaciones por computadora dan facultad al alumno para entrar datos y variables y para manipular los elementos que intervienen en la experiencia modificando el resultado de la experimento.

Las simulaciones pueden referirse a actividades, procesos y fenómenos relacionados con la naturaleza, la ciencia la técnica, la industria, el comercio, etcétera. Gracias a la memoria y velocidad de la computadora y a su capacidad de localizar y visualizar instantáneamente todo tipo de información (numérica, alfanumérica, gráfica, animación), las simulaciones se han extendido desde la primitiva esfera de los sistemas matemáticos, físicos o químicos, a los campos biológico, geológico, astronómico, económico y social entre otros.

Los programas de simulación exploran situaciones presentadas mediante secuencias gráficas, fijas o dinámicas, que evolucionan según la táctica que aplique el alumno para resolver cada caso. El objetivo de

[3] Ibid. pp. 18-23.

esta modalidad de aprendizaje es reflejar la importancia de los distintos factores que intervienen en un determinado proceso y descubrir la incidencia y naturaleza de las condiciones que posibilitan su modificación. Para conseguirlo, el autor de un programa de simulación suele dejar algunas variables libradas a las decisiones que va tomando el estudiante.

La intervención del alumno se da en distintos niveles, según el tipo de programa. A veces, puede implicar la atribución o el cambio de ciertas magnitudes que condicionan la ejecución del experimento (velocidad en el caso de una nave espacial) o bien, la toma de decisiones puede darse a un nivel más preciso como optar por el camino a seguir en una exploración geológica.

Otra variante de programas de simulación es la de aquellos que simulan una actividad real y cotidiana, con frecuencia de carácter profesional. Existen programas de este tipo para aprender a llevar un pequeño comercio o unos grandes almacenes, para diseñar políticas de mercadotecnia y estudiar las consecuencias de determinadas operaciones comerciales, para desempeñar ciertas funciones empresariales (jefe de personal, director de establecimiento, etcétera); estos simuladores de negocios no suelen incorporar efectos gráficos y proponen al alumno una serie de opciones o decisiones alternativas, para que este resuelva cual es la idónea es una situación concreta. Las preguntas suelen estar basadas en el principio de elección múltiple.

En general crear programas de simulación con gráficos requiere tener opciones de programación bastante avanzadas y es preciso dominar un lenguaje de programación como Pascal o Basic. También Logo puede ser apto en determinados casos.

2.2.3 Programas lúdicos y juegos didácticos.

Los juegos educativos tienen mucha aplicación en la IAC y los programas de autor se prestan naturalmente a crear actividades lúdicas, por ejemplo basadas en textos o en listas de palabras para la enseñanza de idiomas. Hay una abundante oferta de juegos educativos para microcomputadoras y muchos ofrecen al profesor o autor la posibilidad de introducir el contenido dentro de un formato de programa fijo.

Existe una estrecha relación entre los programas de juegos y los programas con una presentación más "seria" muchos de los cuales también incluyen actividades o aspectos lúdicos.

Además, varios programas contienen, entre otras actividades, alguna actividad más bien lúdica que, por ejemplo, permite variar el formato del ejercicio para transformarlo en un juego. A veces se ofrece al alumno la posibilidad de hacer un repaso más atractivo del contenido de los ejercicios previamente realizados. Por otra parte, incluso las actividades más "serias", como los ejercicios y prácticas, incluyen a menudo elementos lúdicos, tales como dar al alumno las opciones de determinar aspectos de la actividad como la puntuación, restricciones de tiempo, grados de ayuda a recibir, cantidad de consultas al texto o de controlar aspectos de visualización (color, tipo de letra, etcétera). Esto además da una gran flexibilidad a los ejercicios aumentando la motivación del alumno y brindándole la posibilidad de elegir la manera en que va a participar.

También existen programas de actividades intrínsecamente educativas que a la vez tienen rasgos más bien lúdicos. Un ejemplo serían los programas para el aprendizaje de idiomas basados en la reconstrucción de un texto, con todas o algunas de sus palabras en blanco. En este caso, la capacidad para la predicción o adivinación de las palabras dentro de un contexto específico se ve como un componente de la competencia en el uso de lenguaje y por lo tanto constituye un fin intrínseco de la actividad. Se presupone que la realización de esta actividad constituye un trabajo con valor cognitivo que posibilita profundizar las relaciones entre las reglas y formas aprendidas y el campo de la experiencia y la memoria del alumno.

El elemento lúdico suele convertir un ejercicio aburrido, por ejemplo de elección múltiple o de emparejamiento, es un desafío o entrenamiento motivador. Algunos problemas lúdicos pueden servir para revisar o ampliar temas ya dados como por ejemplo, los programas que permiten crear juegos para reconocer

parejas de elementos "ocultos" (tales como un número y su múltiplo, una palabra y su definición, etcétera.) representados en la pantalla por barajas o cuadros que el alumno debe destapar y relacionar entre sí.

Los programas lúdicos no necesariamente se limitan al campo más obvio del repaso y la práctica. Algunos programas también sirven para realizar actividades de descubrimientos, entendidos como una manera de introducir información o conceptos nuevos. Las actividades de clasificación o emparejamiento (que permiten incluir elementos lúdicos) son aptas para este uso.

Los programas lúdicos pueden fomentar una dinámica de interacción distinta de la que prevalece en la relación típica entre el profesor y el alumno. El alumno considera a la computadora como un adversario al que puede ganar y reclama la ayuda de sus colegas para hacerlo.

Los juegos didácticos a veces creados con programas de autor, se adaptan con facilidad a las metodologías más diversas, a varias situaciones de clase y a diferentes programas de estudios.

Los juegos también pueden tener desventajas. Tanto los alumnos como el profesor pueden confiar demasiado en el funcionamiento automático de las actividades como mecanismo de aprendizaje. Existe el peligro de que las actividades lúdicas centren la atención del alumno en la meta de la actividad a fin de ganar puntos a la computadora o de completar una tarea extrínseca al contenido didáctico y de que los alumnos impliquen su percepción y cognición sólo lo suficiente para apoyar este fin. Para evitar el riesgo de un aprendizaje poco profundo es conveniente que las clases incluyan actividades de consolidación de lo aprendido por medio de la computadora.

2.2.4 Sistemas expertos.

Los sistemas expertos son programas basados en una forma de programación más "inteligente" que la habitual en otros programas de IAC. Un sistema experto tienen a veces la capacidad de "aprender" nuevos datos o relaciones durante la ejecución del programa. El desarrollo de estos programas es tal vez uno de los campos prometedores de la informática educativa y se halla estrechamente relacionado con los avances de la investigación en el campo de la inteligencia artificial (IA).

Los estudios de IA se orientan hacia nuevos modelos de comunicación hombre-máquina. Esto se intenta conseguir introduciendo interfases muy potentes, capaces de procesar la multiplicidad de formas semánticas y sintácticas que una misma instrucción puede tener en un lenguaje.

Para la programación de sistemas expertos pueden utilizarse lenguajes tradicionales de programación, en cuyo caso el profesor o experto trabaja habitualmente en equipo con programadores, o lenguajes de autor bastantes sofisticados. Pero la tendencia actual se orienta principalmente al desarrollo de lenguajes más adaptados para la creación de estos sistemas y más flexibles para crear programas de inteligencia artificial.

2.2.5 Sistemas de diálogo.

Los sistemas de diálogo han sido desarrollados como resultado de avanzadas investigaciones en el campo de la inteligencia artificial, permiten obtener un auténtico diálogo interactivo. Este sistema de diálogo tiene dos modalidades. Cuando la computadora tiene la iniciativa, el programa presenta un cuestionario seleccionado al azar, entre las muchas preguntas que el sistema es capaz de seleccionar a partir de unos datos básicos. Es un sistema generativo que se vale de una lógica propia; normalmente procede por propiedad transitiva.

En la segunda modalidad, cuando el estudiante decide interrogar al programa sobre cualquier información, el sistema de diálogo, mediante sus reglas de producción, puede contestar a preguntas cuyas respuestas no hayan sido previstas. El sistema consigue reconocer las propiedades de los datos que se le suministran y en

consecuencia deduce respuestas lógicas. Este esquema tiene un gran atractivo para aquellos educadores que creen que el mejor uso de la computadora es su utilización plenamente libre como herramienta del alumno.

En principio, LISP, LOGO y PROLOG parecen imponerse como lenguajes para estos usos didácticos, por ser conceptualmente los más cercanos al campo de la inteligencia artificial.

2.2.6 Programas de evaluación.

Por otra parte, los programas de evaluación para los que existen programas de autor de sencillo manejo, no deben verse exclusivamente como instrumentos para reemplazar a los exámenes, aunque a largo plazo también les sea referida probablemente esta función (para aliviar al maestro de la monótona tarea de corregir respuestas perfectamente previsibles, e individualizar el nivel de la evaluación en función de la historia de cada alumno).

Una prueba de elección múltiple puede servir, entre otras cosas, para la comprobación del nivel de los alumnos al comienzo del cursos, para diagnosticar problemas de aprendizaje y planificar la estrategia de seguimiento, para evaluar en grupos la evolución de los alumnos a lo largo del curso (sin carácter punitivo, ni registro de puntos por individuo), etcétera.

El método de evaluación por el sistema de elección múltiple es quizás el más conocido, pero no el único. Otro formato de evaluación sería el de completar palabras o rellenar blancos, muy frecuente en la enseñanza de idiomas. Pero los ejercicios y evaluaciones basados en completar blancos no son solamente evaluaciones, a menos que el profesor desee limitar su uso a este plano.

Varios lenguajes y sistemas de autor pueden utilizarse para realizar cuestionarios y evaluaciones, o autoevaluaciones. En efecto, los lenguajes y sistemas de autor son muy aptos para la creación de esos programas, puesto que los docentes casi han de intervenir en la elaboración del contenido de estas actividades. En general, es muy limitada la posibilidad de utilizar programas de práctica de exámenes estandarizados, o las evaluaciones que acompañan a ciertos cursos basados en libros de texto o en otros soportes.

La autoevaluación permite disminuir la tensión que provoca la situación de examen, al poner los resultados a disposición del alumno para su autocontrol. Varios sistemas y programas de autor ofrecen la posibilidad de programación y consulta de diccionarios, páginas de ayuda, o un cuidadoso seguimiento de las contestaciones del alumno para guiar la construcción de la respuesta esperada (advertencias de error y de faltas ortográficas, ayuda y repaso de las preguntas mal contestadas, etcétera).

Los sistemas y tests para realizar evaluaciones suelen ser de manejo muy sencillo y abundan versiones para microcomputadoras, principalmente para la enseñanza de idiomas.

2.2.7 Tutoriales.

Existen distintas clases de programas tutoriales. Muchos suelen limitarse a presentar nueva información. Otros sirven para ejemplificar o ilustrar conceptos previamente estudiados.

Aunque su estructura es a menudo secuencial, estos programas también ofrecen otras formas de presentación, con índices de actividades o menús de opciones disponibles.

Los programas tutoriales presentan información como un libro, bajo el control del alumno y su ritmo. Se utilizan a menudo para presentar información técnica, relacionada por ejemplo con el uso de un lenguaje o del sistema operativo de una microcomputadora. En este caso, un menú permitirá al estudiante ver la demostración de los aspectos que le interesan. El programa puede incluir algunas interacciones elementales,

tales como interrogar al alumno sobre su comprensión de la presentación, o darle la posibilidad de reforzar su aprendizaje con una simple acción (al presionar una tecla). A veces también es posible determinar algún aspecto de la demostración.

Los programas tutoriales tienen más utilidad para presentar conceptos elementales, e información ya que es indispensable memorizar (información técnica, etcétera) A veces ofrecen una práctica rudimentaria de habilidades específicas, por ejemplo asociadas con el manejo de programas de aplicación. También suelen llamarse programas de demostración los que sirven para demostrar un concepto ya dado en clase, mediante gráficos o animaciones que permiten un mínimo de interacción.

Las posibilidades que ofrece la computadora de incluir representaciones gráficas, con animación o movimiento y de crear efectos sonoros, junto con su gran rapidez y capacidad de cálculo, son recursos a tener en cuenta al evaluar el uso de lenguajes y sistemas de autor para crear tutoriales.

La mayoría de estas aplicaciones son aptas para ser usadas directamente por el profesor al frente de la clase, a modo de pizarrón electrónico.

2.3 Modelos de uso de la computadora.

Las primeras concepciones de la IAC se apoyaban en un modelo de aprendizaje individualizado, que permitiera al alumno progresar a su propio ritmo. En este esquema el contenido de las lecciones se limitaba a la presentación de información y a la subsecuente comprobación de su memorización por el alumno. La computadora evaluaba las respuestas del alumno, las confirmaba y corregía, o simplemente repetía la información no aprendida. Actualmente este modelo skinneriano no goza de mucho apoyo explícito, por lo menos a nivel teórico, pero su influencia se manifiesta implícitamente en muchos programas actuales. Su persistencia en la IAC puede deberse más a pereza y falta de creatividad por parte de los diseñadores de programas, que a la aceptación general del modelo conductista.

Hoy en día, con la entrada de la microcomputadora en la escuela, este modelo casi exclusivo ha dejado paso a unos métodos más innovadores para el uso de la computadora. A continuación se consideran los diferentes modelos de trabajo más frecuentes que E. Giordano y R. Edelstein señalan según la disposición física de los alumnos ante la computadora. [4]

2.3.1 Modelo individual.

El modelo de trabajo individual puede ser compatible con metodologías más actuales, aunque su papel en estas se ve reducido. Ante todo destaca el concepto de acceso optativo *self-access*, que es muy corriente en la enseñanza de idiomas. Se trata de asegurar que el alumno disponga de recursos para el trabajo individual, a menudo fuera de la clase, que puedan motivarlo y darle sentido un sentido de responsabilidad al aprendizaje, teniendo en cuenta las necesidades particulares de cada estudiante.

2.3.2 Grupos reducidos.

El uso de la computadora en pequeños grupos de alumnos es más económico, pero lo más importante es la dinámica de interacción que surge entre los alumnos, al estimularse entre sí para encontrar una respuesta o solucionar un problemas. En ciertas actividades, los alumnos que trabajan en grupo experimentan una sensación de solidaridad frente a la máquina y le intentan "ganar". A veces los alumnos saben más que la computadora y al percibir que ésta no es tan omnipotente como el profesor, adoptan un papel menos pasivo. El programa puede servir como estímulo para la discusión, permitiendo socializar el aprendizaje.

---

4 Ibid. pp. 25-26.

Trabajo integrado.

El aprendizaje mediante programas didácticos conlleva el peligro de que los alumnos se centren demasiado en las tareas y objetivos extrínsecos que les propone el programa, en lugar de explotar todas las posibilidades que ofrece el contenido didáctico. En algunos casos los alumnos pueden limitarse a ganar puntos, o solamente a teclear la palabra o respuesta adecuada. Dejar un espacio en el aula para trabajar sin computadoras permite la planificación de actividades anteriores o posteriores a la sesión de trabajo con la computadora, para plantear aspectos previos a la actividad, comprobar la profundidad y eficacia del aprendizaje y transferir las experiencias ganadas del mundo externo a la pantalla. Este modelo de uso de la computadora permite consolidar y extender un aprendizaje poco profundo, convirtiéndolo en una experiencia útil y eficaz.

Pizarrón electrónico.

Bajo este modelo, la computadora se sitúa al frente de la clase y sirve para la presentación de material gráfico o textual. A menudo se utiliza este modelo con programas de demostración o simulación, aunque no excluye la presentación de ejercicios u otras actividades. El profesor puede mostrar imágenes, gráficos, o textos como en un pizarrón. La capacidad de animación que tiene la computadora también le permite mostrar procesos tales como experimentos científicos, de física, química, biología, etcétera. Este uso de computadora admite además la posibilidad de interrumpir un proceso, o de cambiar sus parámetros determinantes, convirtiéndolo en una demostración interactiva.

Recurso.

Dentro de este esquema, una computadora instalada en el aula sirve como recurso o estímulo para realizar actividades en grupo. En medio de una determinada actividad, un alumno del grupo accede a la computadora para buscar información y llevarla después a sus compañeros. El programa puede consistir, por ejemplo, en una simulación que responda a decisiones tomadas por el grupo tras una discusión previa y puede ayudar al grupo a evaluar las consecuencias de sus decisiones y corregirlas, facilitando un resultado o nueva información. Asimismo, las bases de datos y comunicaciones por vía telemática sirven también como recurso para una actividad de grupo.

2.4 Programas educativos lineales y ramificados.

La enseñanza programada (EP) representa un estilo de programa lineal, donde el estudiante avanza lección por lección y unidad por unidad, siguiendo un camino ascendente de niveles de dificultad previamente fijados por el autor del curso. La sola idea de curso es representativa de esta clase de programas, por el carácter rígido que denota este método. Todo el material se presenta en secuencias lineales, atomizado para su memorización a corto plazo y no se ofrece al alumno la posibilidad de optar por los contenidos más sugerentes para él. [5]

Los llamados programas ramificados no son muy diferentes de la enseñanza programada lineal aunque presentan un itinerario más atractivo. La primitiva EP de Skinner modificó algunas de sus técnicas tras las investigaciones del también psicólogo Norm Crowder, quien desarrolló el concepto de programas ramificados. En estos programas el estudiante recorre un camino principal, del que a veces sale por ramificaciones que determinan sus propias respuestas, dirigiéndose por una u otra bifurcación, para volver luego al programa secuencial. Muchos lenguajes de autor han sido diseñados para poder crear programas ramificados de EP.

---

5 Ibid. pp. 32-33.

Generalmente se reconoce que esta clase de programas tienen la virtud de presentar los contenidos de un modo más articulado que la EP lineal, donde la sucesión de pantallas es invariable. Los programas ramificados permiten individualizar mejor el aprendizaje, comentando posibles errores, incorrecciones o aciertos (feedback o retroalimentación), pero su concepción no difiere esencialmente del modelo lineal en sus principales defectos: su evolución en forma de curso (no controlada por el profesor o el alumno) y su inexistencia en la mera transmisión y memorización de contenidos al modo tradicional.

2.5 Desarrollo de material curricular para computadora.

Crear material de curriculum de calidad es un trabajo serio y difícil, independiente de la cuestión del medio que se emplee.

2.5.1 Elaboración del curriculum.

Alfred Bork [6] señala que la forma como se producen los libros puede servir como introducción general para la preparación de material de curriculum, ya que concurren muchas características iguales o parecidas en la elaboración de este tipo de material incluyendo al que implica a las computadoras.

Es conveniente establecer el proceso en cinco etapas a tenerse en cuenta:

1. Diseño pedagógico.
2. Diseño gráfico.
3. Realización.
4. Evaluación.
5. Perfeccionamiento.

Diseño pedagógico.

El diseño pedagógico se refiere a todo el proceso de planificación del material, la planificación que precede a cualquier realización.

El diseño pedagógico normalmente tiene varias etapas. Primero incluye el plan inicial del material para la lección, el desarrollo de la estrategia en conjunto antes de entrar en detalles. Esta estrategia de conjunto puede surgir de una variedad de métodos. Normalmente en este plan inicial se consideran los fines y los objetivos. Una técnica útil es la de lluvia de ideas cuyo objetivo primordial es el de que surjan muchas ideas para la próxima etapa del proceso.

Diseño gráfico.

La segunda etapa en la elaboración del proceso es el diseño gráfico, que esta relacionado con la aparición visual y posiblemente temporal del material, su estructura en el espacio y el tiempo.

Realización.

La producción de los módulos didácticos es el aspecto más técnico de la realización, que casi siempre requiere el trabajo de una serie de especialistas. Así, para la realización técnica de las publicaciones están

---

[6] Bork, Alfred. El Ordenador en la Enseñanza (Análisis y Perspectivas de Futuro), España, Gustavo Gili, 1987, pp. 191-225.

implicados tipógrafos, impresores y encuadernadores, mientras que para la realización de películas se precisa de cámaras, directores, equipos de luces, maquilladores, trabajadores de laboratorio que revelan las películas, editores de película, actores, etcétera.

La actual actividad técnica puede ser completamente distinta según los diferentes medios, pero en todos los casos la realización técnica pasa desde el diseño hasta el material de trabajo.

Evaluación y perfeccionamiento.

Las tres etapas que acabamos de considerar - diseño pedagógico, diseño gráfico y realización - son casi secuenciales en la mayoría de los medios aunque puede ocurrir que algunas vayan de una parte a otra, es decir, se puede hacer algún diseño gráfico en la etapa de diseño pedagógico y se puede añadir algún diseño pedagógico después del diseño gráfico. Es posible que las etapas de evaluación y perfeccionamiento se realicen en varias partes a lo largo del proceso de producción y puede que se repita varias veces. Personal competente revisa el guión muchas veces durante su desarrollo. Normalmente, inmediatamente después de cada etapa de diseño pedagógico y también después del diseño gráfico, se produce un ciclo de evaluación perfeccionamiento.

En muchos tipos de diseño instrumental es posible que se haga un ciclo evaluación-perfeccionamiento después de la realización. Su naturaleza puede diferir. La cuestión de cuanto se puede perfeccionar después de la realización depende del medio. Con las computadoras se puede hacer mucho en esta etapa, ya que después de la producción el medio es muy flexible.

2.5.2 Estrategia de producción para computadoras.

A. Bork [7] también señala que aunque los detalles pueden ser distintos, su opinión es que el método más efectivo para utilizarlo es el mismo que se sigue en la producción de material de aprendizaje.

La estrategia que el propone es la que se ha utilizado durante muchos años en el Centro de Tecnología Educacional de California, [8] ya ésta es muy parecida a la estrategia general para el desarrollo de cualquier material de curriculum revisado anteriormente dado que ésta incluye las mismas etapas: diseño pedagógico, diseño gráfico, realización, evaluación y perfeccionamiento.

Diseño pedagógico inicial para material de aprendizaje basado en computadora.

En esta etapa se desarrollaran explicaciones para las unidades del curriculum. Las lluvias de ideas normalmente son muy útiles. En el diseño inicial se debería incluir también la ayuda de personas que hubieran estado directamente relacionadas con trabajos de investigación asociados con el aprendizaje de una asignatura concreta.

El producto de esta etapa será una serie de descripciones de módulos de computadora, basados en el análisis de las necesidades de los alumnos. Estas descripciones pueden ser únicamente de una o de pocas páginas de extensión, pero proporcionarán suficientes detalles para guiarnos hacia una razonable visión de lo que se tiene que hacer. O sea, el resultado del diseño inicial prepara el comienzo de la etapa siguiente.

---

7 Ibid. pp. 197.

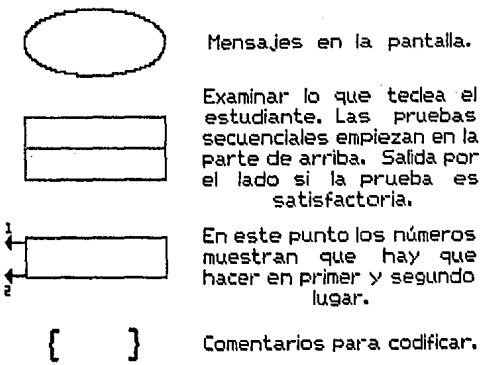
8 Centro de Tecnología Educacional de la Universidad de California, Irvine, E.U.A.

Diseño pedagógico detallado.

El diseño pedagógico detallado es quizá la etapa más crítica y fundamental del proceso de producción, ya que es en la especificación del diseño pedagógico completa donde se determina la calidad de los materiales.

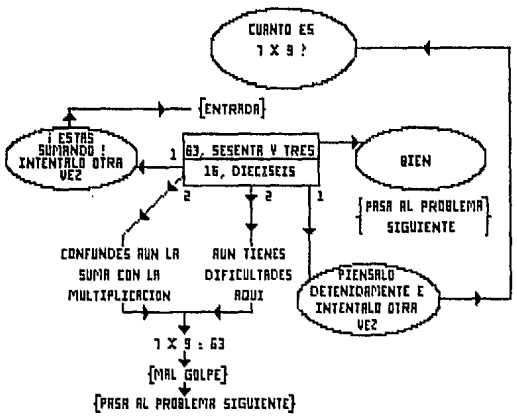
En esta parte se trabaja íntimamente en el tema a tratar cuyo propósito es el de producir el guión para un nuevo módulo de aprendizaje, así como la entrada a posteriores etapas del proceso de producción. El guión debe dar todos los detalles de lo que aparecerá en la pantalla (tanto textos como dibujos), como aparecerá, como analizar la entrada del alumno y que decisiones se habrán de tomar en base a este análisis. La figura 2.5.1 explica las convenciones sencillas para guiones de computadora.

Figura 2.5.1 Convenciones del guión



La figura 2.5.2 presenta un guión de muestra en una forma utilizada durante muchos años en el Centro de Tecnología Educacional de California el cual visualiza las tácticas para la producción de guiones. El mensaje "¿cuanto es 7 X 9 ?" se visualizará en la pantalla de la computadora. El material en corchetes, tal como "entrada" proporciona instrucciones al codificador en la etapa de realización, o comentarios dentro del programa para hacerlo más legible. Durante la codificación quizás se puedan necesitar varias instrucciones por parte de los autores y por lo tanto estos corchetes sirven para separar estas instrucciones de otros detalles especificados; dichos corchetes son utilizados en el Centro de Tecnología Educacional, como símbolos de "comentario" y han sido adoptados por A. Bork para los guiones.

Figura 2.5.2 Ejemplo de un guión



Las cajas que están situadas en el centro del gráfico son los tests en la entrada de los alumnos. Estos tests aparecen en secuencias, lo cual quiere decir, que si el primer test en la lista no va bien, se hace el segundo. Si este tampoco tienen éxito se hace el tercero y así sucesivamente. Si el test tiene éxito e implica que la serie que se está poniendo a prueba está contenida en lo que el alumno ha tecleado, entonces se deja la caja por la parte derecha o por la izquierda a lo largo de la línea que muestra salida en aquella dirección. Así, en la muestra de la figura 2.5.2, si el alumno teclea la respuesta correcta, 63, la computadora contestará "bien" y pasará al problema siguiente.

Un importante aspecto pedagógico a tener en cuenta es que, además de buscar los errores que es probable que cometan los alumnos, debemos considerar las respuestas por anticipado. Al ser capaz de dar un refuerzo inmediato en tales situaciones, un aspecto importante del aprendizaje basado en computadoras hace que éste sea un aparato didáctico eficaz en clases normales con un elevado número de alumnos es difícil que los profesores puedan ofrecer una ayuda tan rápida.

Dentro del diseño pedagógico se debe determinar lo que hay que hacer en cada caso, incluyendo el caso de la respuesta imprevisible.

Cada decisión que sea tomada por el alumno deberá basarse en las necesidades pedagógicas del momento. Así, si la cuestión es muy importante, probablemente desearemos dar un ayuda amplia, o hacer la pregunta de otra manera o escoger otras estrategias para ayudar a aprender. Por otra parte, si se trata de algo trivial desearemos dar la respuesta inmediatamente y seguir con el material.

Diseño de pantalla.

La siguiente etapa en el desarrollo del material de aprendizaje basado en computadora es la de diseñar el modo en que aparecerá éste en la pantalla, tanto en el aspecto espacial, es decir, la colocación de textos y gráficos, como en el aspecto temporal, es decir, el tiempo de aparición del material en la pantalla.

Debemos ser conscientes de las posibilidades del diseño de pantalla de manera que la podamos utilizar correctamente tomando en cuenta que pueden ser empleados algunos diseños de pantallas ya editados anteriormente. En cada etapa necesitamos preguntarnos como la información a través de la pantalla puede contribuir a ayudar en el aprendizaje.

Durante el diseño de las pantallas se debe trabajar de manera interactiva, directamente sobre la pantalla de la computadora, colocando los objetos y los textos, trasladándolos y cambiando su orientación hasta adquirir la forma adecuada.

Para el buen manejo de la pantalla pueden seguirse los principios generales en cuanto a espacio y tiempo. De todos estos principios el más importante sea tener en cualquier momento la mínima cantidad de texto o de gráficos en la pantalla.

Por lo tanto, se debería extremar en lo posible el espacio vacío en la pantalla. La manera para conseguirlo es visualizando una información mínima en cada momento y quitar el texto y los dibujos que no sean relevantes. Las pantallas visualizadoras modernas permiten eliminar algún material sin borrar todo lo que aparece en ella y por lo tanto permiten una información fluida.

Existen otros factores que pueden ayudar a la legibilidad. Las líneas cortas, el evitar escribir con guiones para unir o separar, paréntesis, eliminar justificaciones al margen, tanto a la derecha como a la izquierda, mantener las frases naturales juntas en una línea, hacer pausas antes o después de palabras o frases críticas y disminuir la velocidad de presentación de un texto hasta conseguir un ritmo agradable para el usuario medio. Todo esto contribuye a que el texto sea más legible y útil. Los textos y los gráficos deberían funcionar conjuntamente para reforzar la información.

Codificación.

El proceso siguiente es escribir el programa para producir el código.

La codificación se basará principalmente en la lógica del proceso interactivo. Es decir, programas que analicen las entradas de los alumnos y que toman decisiones apoyándose en estos análisis. Durante la etapa del diseño pedagógico este análisis se ha especificado en el guión y con los productos de diseño gráfico.

Las características de programación estructurada deberán ser tomadas en cuenta en el momento de escribir los códigos para los diálogos didácticos basados en la computadora. Los diálogos deben tener los mínimos errores posibles; el código deberá ser revisado por medio de sucesivas revisiones (algunas inmediatamente y otras posteriormente), este deberá ser mantenible cuando se descubran errores difíciles y deberá ser trasladable (fácil de trasladar a otras máquinas).

Es importante que antes de empezar a trabajar en el programa planificar el código.

También es importante considerar el lenguaje de programación y las condiciones en que se introduce con mucho cuidado. Las condiciones incluyen al editor y otras ayudas de programación. La codificación compleja moderna se hace relativamente, en pocos lenguajes.

El aspecto más delicado en cuanto a lenguajes de computadoras bajo el punto de vista estándar del material de aprendizaje basado en computadoras es el diseño de pantalla, la habilidad de situar el material en la pantalla en un tiempo determinado. Los requerimientos gráficos son apropiados, ya que son los mejores sistemas de manejo de texto en la pantalla.

El código debe satisfacer dos requerimientos: debe hacer lo que se pretende que haga y debe ser un código bien estructurado.

Análisis, evaluación y revisión.

Uno de los aspectos más importantes del material didáctico basado en computadoras es que pueden aparecer el análisis y la evaluación en muchas etapas a lo largo del proceso, más que en cualquier otro medio.

Los evaluadores pueden disponer de una visión muy detallada del material en cada punto -una microvisión- de lo que ocurre con un alumno, ya que un programa de aprendizaje basado en computadora que sea bueno es muy interactivo.

El análisis, la evaluación y la modificación del material pueden tener lugar al menos en nueve puntos durante el proceso de producción. Además, para cada etapa, son posibles varios ciclos de análisis y evaluación seguidos de perfeccionamiento. Los posibles ciclos de análisis, evaluación y perfeccionamiento son:

Análisis del diseño de especificación.
Análisis del diseño pedagógico detallado tal como se especifica en el guión.
Análisis del diseño de pantalla.
Análisis del diseño de codificación.
Análisis de la versión de funcionamiento del diálogo de computadoras.
Análisis del código de computadora.
Evaluación formativa de la versión de funcionamiento.
Evaluación en grupo de la versión de funcionamiento.
Evaluación sumativa del producto final.

Para la mayoría de estas etapas debe hacerse una serie de preguntas. Algunas de estas preguntas se repiten en varias ocasiones. Los grupos implicados en el análisis y la evaluación diferirán de un caso a otro, según varios actores tales como el público a quien van destinados y los fondos disponibles.

Análisis de la especificación del diseño.

La primera etapa en el proceso de desarrollo es la especificación de diseño. Una o varias páginas para describir cada uno de los módulos a desarrollar. La especificación de diseño es un documento escrito. Puede enviarse a varias personas para un análisis breve o un análisis detallado. Se pueden plantear las siguientes preguntas:

- El diseño, ¿está basado en las investigaciones realizadas?
- El diseño, ¿es claro?
- El diseño, ¿proporciona entradas adecuadas para el grupo de diseño detallado?
- El diseño, ¿es práctico para el uso de un alumno?
- El diseño ¿es consecuente con los sistemas de distribución?
- El diseño, ¿hace uso de las capacidades de la computadora?
- ¿Hay otros medios posibles?

Análisis del guión.

El producto de los grupos de diseño detallado es el guión. Los guiones pueden también enviarse para ser analizados puesto que son productos de papel.

Unas cuantas preguntas a hacer en esta etapa incluyen:

- ¿Es claro el texto o pueden mejorarse los mensajes?
- ¿Es conciso?
- El texto, ¿es apropiado en cuanto a nivel y vocabulario para el público a quien va dirigido? (Esto implica saber para quien se ha elaborado el material).
- Los mensajes, el texto, ¿son agradables y útiles para el usuario?
- ¿Puede la información visual adicional ayudar a los alumnos? (Debería ponerse un énfasis especial a esta cuestión, ya que los maestros muchas veces no pueden proporcionar una suficiente información visual relevante).
- ¿Se pueden mejorar las disposiciones de la pantalla?
- ¿Hay errores de lógica en el material que pudieran hacer imposible llevar a cabo el código tal como se indica en la pantalla?
- ¿Se han eliminado bucles infinitos que hacen caer en la trampa al estudiante, excepto los bucles triviales dejados intencionadamente?
- ¿El análisis de la entrada del alumno es apropiada para el público a quien va dirigido?
- ¿Hay respuestas correctas que no se han indicado en el guión?
- ¿Hay respuestas erróneas o incorrectas que el programa debería estar buscando pero que fueron omitidas en la creación? (Son importantes las respuestas de los alumnos que sugieren un reforzamiento útil inmediato).
- ¿Se pueden sugerir secuencias de ayuda adicional o de mejoramiento en las secuencias de ayuda ya propuestas? (La eficacia del material de aprendizaje basado en la computadora depende de si es capaz de dar la ayuda adecuada a todos los alumnos que tengan dificultades).
- ¿Puede mejorarse la interacción entre computadora y estudiante?
- ¿Se han descuidado determinados casos indicados en el guión por líneas que no conectan con nada o por líneas que se han perdido?

Análisis del diseño de pantalla.

Si el diseño de pantalla se hace de forma interactiva con los resultados de diseño disponibles solamente con la computadora, el proceso de revisión debe tener lugar en las pantallas del visualizador.

Aquí se exponen las preguntas que se pueden hacer:

- Las pantallas ¿Son atractivas para el público a quien van destinadas?
- ¿Se ha tomado en cuenta dejar el máximo espacio vacío posible?
- ¿Se han suprimido informaciones innecesarias manteniendo las pantallas "limpias"?
- ¿Se ayuda a la legibilidad para el estudiante?
- Las líneas de texto, ¿son cortas?
- Las frases naturales, ¿se mantienen juntas en las líneas de texto?
- ¿Se utilizan las pautas después de una puntuación principal?
- ¿Hay pausas para aislar palabras y frases importantes?
- ¿Es posible más información visual adicional o pictórica?
- ¿Es conveniente la información adicional, visual o pictórica?
- El diseño, ¿hace uso de los métodos de intensificación como el parpadeo y el video con marcha atrás?

Análisis del diseño de código.

En un proyecto de codificación amplio tal como los asociados con materiales de aprendizaje basado en la computadora, es inconveniente diseñar el código antes de que se escriba. Este diseño podría hacerse en forma de esquemas estructurados u otros aparatos surgidos de la ingeniería moderna del software.

Los expertos en software son los analistas más convenientes en esta etapa. Aquí se exponen las preguntas que se pueden hacer:

- El diseño de código, ¿es modular?
- Cada uno de los procedimientos o módulos, ¿son de una longitud razonable?
- El papel de cada módulo, ¿esta definido de una forma clara dentro de la estructura de diseño?
- Los nombres de los módulos son indicativos de sus funciones?
- Las estructuras de los datos, ¿están claramente especificadas?
- ¿Está claro que datos hay que suministrar a cada módulo?
- ¿Está claro cuales datos son los globales para el programa entero y cuales datos están restringidos a ciertas partes del programa?

Análisis interno de los diálogos en funcionamiento.

La disponibilidad del primer material en funcionamiento es una etapa emocionante, un punto importante en el desarrollo de los materiales de aprendizaje basado en computadoras. Se puede hacer funcionar el programa muchas veces, tomando nota de las cosas que necesitan mejorarse. Este análisis interno y el perfeccionamiento es muy probable que ocurran muchas veces antes de que el diálogo se considere listo para otras pruebas.

En esta etapa se pueden plantear las siguientes preguntas:

- El programa, ¿funciona en todas las situaciones? (Esto podría determinarse hasta cierto punto, haciéndolo funcionar muchas veces y probando muchas entradas diferentes. Puede ser que se desee introducir las entradas correctas, pero también es necesario probar todas las secuencias de ayuda).
- Los gráficos, ¿son adecuados?

- En el proceso de aprendizaje, ¿sería de ayuda una información visual adicional?
- ¿Puede mejorarse el diseño de pantalla?
- ¿Puede mejorarse la interacción?

Aunque se hace un listado de pocas preguntas, cada pregunta realizada en las etapas anteriores podría plantearse durante el proceso de revisión interna.

Análisis del programa.

Esta etapa es la segunda en importancia del proceso de análisis del software necesaria para asegurarse de que el código es legible, modificable y transportable. Aunque se sitúa esta después del análisis del programa en funcionamiento, ambas pueden hacer simultáneamente. Los principios de análisis son los de la ingeniería de software moderna. Las preguntas coinciden en parte con las planteadas anteriormente en cuanto al diseño de código se refiere.

- El programa de computadora, ¿es legible? es decir, cualquiera que no sepa nada relacionado con programación, ¿puede descubrir lo que hace el programa y como lo hace?
- ¿Es fácil entender la especificación de los datos?
- Los identificadores, ¿son razonables? (Teniendo en cuenta los nombres de los procedimientos, los tipos de datos y las variables), ¿es posible tener alguna idea de lo que aquellas entidades hacen o representan dentro de la estructura del programa?
- ¿Es posible sugerir nombres más apropiados?
- Los procedimientos - los módulos individuales-, ¿son de una medida razonable?
- El programa principal, ¿muestra claramente la estructura del programa completo?
- ¿Hay suficientes comentarios para que un lector nuevo conozca lo que hace el programa principal y lo que hacen los procedimientos?
- ¿Hay otros comentarios que puedan mejorar la legibilidad del programa?

Algunas características que dependen de una computadora concreta claramente únicas, ¿están descritas como comentarios adecuados que permitan que alguien los pueda reproducir en una computadora distinta? Esto es importante a tener en cuenta para su compatibilidad. Algunas características que dependen del sistema tales como gráficos, detalles del sistema operativo o del código de unión pueden ser esenciales. Estas características que dependen del sistema deberían mantenerse hasta un mínimo y ser descritas cuidadosamente con comentarios concretos.

Evaluación formativa e interpretación de funcionamiento.

La evaluación formativa incluye a muchos de los alumnos del público a quien ira destinado. Las preguntas a plantear son similares a las asociadas con el análisis interno del programa en funcionamiento. Un nuevo factor importante es que el alumno aprenda utilizando el material.

Se pueden obtener varios tipos de información durante el proceso de evaluación formativa. Parte de la información se almacena directamente por medio de la propia computadora haciendo así uso de los datos amplios de esta para reunir actitudes, las cuales son incomparables en cualquier otro medio de actividades didácticas.

Es posible aprender mucho acerca de como funciona un programa por medio de sesiones de video y por evaluaciones en grupos, evaluación que es utilizada por expertos en la enseñanza de una área y que tienen una experiencia considerable con los alumnos.

Cuando los alumnos utilizan material de aprendizaje basado en computadoras, éste puede guardar varios tipos de información acerca del progreso individual de los alumnos por medio del programa, y también puede generar información estadística de grupos de alumnos.

Tenemos la posibilidad de guardar cada respuesta del alumno a una pregunta dada, o guardar todas las seleccionadas. Sin embargo, es mejor sugerir desde el principio dentro del guión que respuestas deberían guardarse.

Cada vez que se guarda este tipo de respuestas, debe identificarse con la pregunta concreta. También es conveniente permitir que los usuarios hagan comentarios, tal vez indicando los lugares en el programa que ellos consideran confusos o áreas que no son de su interés así mismo, es conveniente guardar los datos de las trayectorias seguidas por los alumnos individualmente o por el grupo entero que se esta examinando. Estos mapas que indican el progreso de los alumnos pueden mostrar donde es necesario que se refuerce el programa. Por ejemplo, si la mayoría de los alumnos terminan utilizando muchas secuencias de ayuda y parece que pocos han utilizado los módulos, es señal de que las secuencias deben mejorarse.

Por otro lado los tests que se incluyen dentro del programa deben elaborarse cuidadosamente teniendo en cuenta los objetivos del material didáctico. Los tests previos y posteriores muchas veces son muy útiles para estructurar la experiencia didáctica dentro de un diálogo, además de obtener información acerca de como se podría mejorar dicho diálogo.

Además de la información almacenada, los evaluadores podrán trabajar directamente con los alumnos en el proceso de la evaluación formativa. Pueden, por ejemplo, observar la actuación del alumno estando a su lado o revisando los monitores que estén en otra parte como auxiliares. Estos monitores pantalla tipo televisión muestran exactamente lo que está viendo y tecleando el alumno. Los evaluadores preparados pueden tomar nota de lo que está sucediendo y utilizarlo para reformar posteriormente el programa.

Los evaluadores también pueden ser los responsables de los tests previos y de los tests posteriores en el caso de que no estén incorporados en el programa. Además los evaluadores pueden entrevistar a algunos alumnos que hayan utilizado el material centrándose en los aspectos afectivos y motivadores. Es preciso establecer una norma de modo que la entrevista sea algo más que una serie de preguntas fortuitas.
Evaluación minuciosa

Otro tipo de estudio de evaluación puede estar en relación simultánea con la evaluación formativa de los alumnos. Profesores competentes pueden considerar el material desde una serie de historiales académicos distintos y hacer sugerencias para mejorarlo. Normalmente a este tipo de evaluación se le denomina evaluación minuciosa.

Además de mejorar el material para el uso del alumno este método producirá unidades agradables para los maestros, unas unidades que los profesores preferirán asignar y por lo tanto harán que probablemente los alumnos quieran utilizar este material didáctico en clase. Este tipo de evaluación es más sencilla y menos costosa que la evaluación formativa con grupos de alumnos bastantes numerosos. Si algunos profesores son muy perceptivos, su información puede llevar a mejorar el material.

La evaluación minuciosa es valiosa pero no debe ser la única estrategia a seguir. Utilizar al alumno durante la evaluación es algo muy delicado.

Evaluación sumativa.

La cuestión más importante en relación a la evaluación sumativa del material didáctico basado en computadora es si la evaluación se sustenta en una serie de objetivos explícitos para ver si el material logra lo que el diseñador se propuso conseguir o si la evaluación no tiene objetivos concretos. En una evaluación sin objetivos concretos los evaluadores intentan descubrir los efectos de utilizar las unidades del curriculum.

Pueden desarrollar efectos inesperados pero importantes sin tener nada que ver con el intento original del creador. Este tipo de evaluación se hace muy pocas veces, pero puede deparar información de gran utilidad.

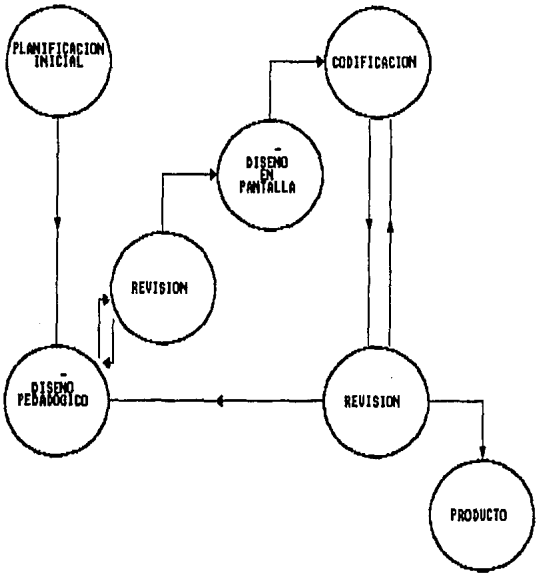
Las evaluaciones sumativas pueden comparar varios métodos para aprender asignaturas concretas. De este modo la elección de los métodos por parte de los educadores será más fácil si saben que los módulos basados en computadoras para solventar problemas de aprendizaje son superiores a otros métodos con clases generales y con profesores corrientes. Comparar los estudios tiene sentido solo si muchas personas están involucradas en excluir variaciones fortuitas.

Dada la cantidad necesaria de alumnos, no es sorprendente que las evaluaciones sumativas sean costosas. Por lo tanto raramente se hacen a una escala adecuada.

2.5.3 El proceso de producción.

Para finalizar la explicación del desarrollo de material curricular en computadora nos podría ayudar un diagrama que mostrará algunas de las etapas. Cada diagrama de este tipo tiende a ser parcial, reflejando algunos aspectos del trabajo pero sin mencionar otros. La figura 2.5.3 muestra las etapas de mayor producción y dos etapas de análisis indicando por medio de flechas que van hacia atrás qué proceso de perfeccionamiento está implicado.

Figura 2.5.3 El proceso de producción de material curricular



3. ANALISIS Y DISEÑO DEL SISTEMA PROPUESTO.

3.1 Requerimientos preliminares

El sistema tutorial que se sugiere como apoyo educativo ha sido propuesto para que el alumno de nuevo ingreso a la Facultad de Contaduría y Administración con los conocimientos mínimos indispensables de computación pueda utilizarlo sin mayor asesoría a fin de ser empleado como apoyo docente en la asignatura de Matemáticas I. Proporcionando al alumno el conocimiento en forma de comentarios y realizando una serie de preguntas y ejercicios pertinentes para reforzar el conocimiento recién adquirido; los ejercicios son generados de manera aleatoria a fin de permitirle al alumno realizar ejercicios diferentes, cada vez que el curso sea estudiado por él. Asimismo se contempla que el sistema tutorial registre información de cada alumno tal como:

- Número de lista.
- Número de cuenta.
- Nombre completo.
- Evaluación por tema y subtema

Esta información se podrá actualizar, dar de baja o consultar.

3.2 Diseño del prototipo preliminar

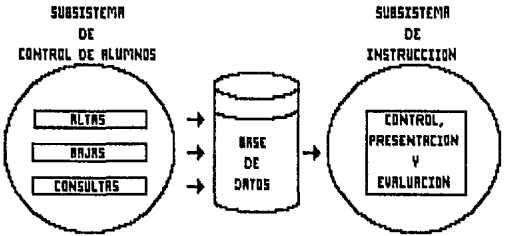
Con la finalidad de lograr una descripción detallada de los requerimientos preliminares del sistema tutorial como el ya descrito y ante la inexistencia de algún modelo computarizado que sirviera como base. Se optó por la elaboración de un prototipo, como parte de la metodología, que permitiera detectar las funciones deseadas en el sistema, la prioridad y frecuencia de cada función, el detalle de los datos a manipularse y la interfase entre la aplicación y el alumno.

Este prototipo sirvió para explicar cada una de las diversas capacidades de proceso que no son visualizadas desde el inicio. Sin embargo son factibles de desarrollarse con la utilización de la computadora dado a que la información accesada por el sistema tutorial se puede almacenar en una base de datos que permita crear un control eficaz y un seguimiento confiable de la información evaluativa de cada uno de los alumnos.

Se utilizó un prototipo debido a que estos se basan en el concepto de sistema amigable y se fundamentan en la utilidad de crear un prototipo de software a construir; tomando la forma de prototipo en papel o prototipo funcional. Para este último, son desarrolladas las funciones generales perfiladas en los requerimientos preliminares, enfocándose a esos aspectos que son visibles por el usuario tales como métodos de entrada, pantallas de captura y otros. Una vez construido el prototipo, se inicia un proceso de interacción mediante el cual se logran obtener los detalles del software a desarrollar facilitando de esta manera la comprensión de lo que el sistema tutorial deberá realizar.

El prototipo que se eligió para realizar el sistema tutorial es el anteriormente descrito llamado prototipo funcional dentro del cual inicialmente se contempló que el sistema tutorial estuviera conformado por tres partes componentes: el Subsistema de Control de Alumnos, el Subsistema de Instrucción y una base de datos (figura 3.2.1).

Figura 3.2.1 Prototipo del sistema tutorial



El Subsistema de Control de Alumnos incluye en su menú principal las funciones de los procesos altas, bajas y consultas, (figura 3.2.2), incluyéndose también las pantallas de captura y las pantallas que registran los datos de identificación del alumno y su avance evaluativo por tema o subtema (pantallas 3.2.3 a 3.2.10).

Figura 3.2.2 Prototipo del Subsistema de Control de Alumnos

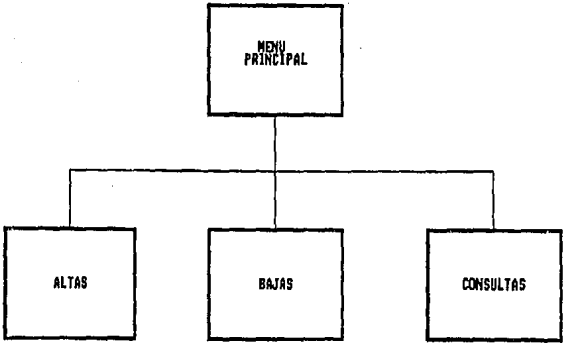


Figura 3.2.3 Pantalla de alta para alumno (prototipo)

ALTA ALUMNO

TECLEE EL NO. DE LISTA DEL ALUMNO :

TECLEE EL NO. DE CUENTA DEL ALUMNO : (OMITA EL GUION)

TECLEE EL APELLIDO PATERNO DEL ALUMNO :

TECLEE EL APELLIDO MATERNO DEL ALUMNO :

TECLEE EL NOMBRE DEL ALUMNO :

(DESEA CONTINUAR EN ALTAS S = SI, N= NO)

Figura 3.2.4 Pantalla de baja para alumno (prototipo)

BAJA GRUPO

ESTA SEGURO QUE DESEA DAR DE BAJA AL GRUPO

TECLEE S > SI , O N > NO:

Figura 3.2.5 Pantalla de baja por nombre para alumno (prototipo)

BAJA ALUMNO

TECLEE EL APELLIDO PATERNO DEL ALUMNO :

TECLEE EL APELLIDO MATERNO DEL ALUMNO :

TECLEE EL NOMBRE DEL ALUMNO :

Figura 3.2.6 Pantalla de baja por número de lista para alumno (prototipo)

BAJA ALUMNO

TECLEE EL NO. DE LISTA DEL ALUMNO QUE DESEA BORRAR:

Figura 3.2.7 Pantalla de baja por número de cuenta para alumno (prototipo)

BAJA ALUMNO

TECLEE EL NO. DE CUENTA DEL ALUMNO QUE DESEA BORRAR:

Figura 3.2.8 Pantalla de consulta para alumno o grupo (prototipo)

PAG. 1

CONSULTAR ALUMNO

NOMBRE DE ALUMNO :
NO. DE LISTA :
NO. DE CUENTA :

LECCION 1.- INTRODUCCION LA LOGICA MATEMATICA

DEFINICION DE LOGICA MATEMATICA	: NP
METODOS LOGICOS	: NP
LENGUAJE SIMBOLICO	: NP

Figura 3.2.9 Pantalla de consulta para alumno o grupo (prototipo)

PAG. 2

CONSULTAR ALUMNO

NOMBRE DE ALUMNO:
NO. DE LISTA :
NO. DE CUENTA :

LECCION 2.- PROPOSICIONES

DEFINICION DE PROPOSICIONES	: NP
SIMPLES COMPUESTAS	: NP
CONECTIVOS LOGICOS Y TABLAS DE VERDAD	: NP
DEFINICION DE CONECTIVOS LOGICOS	: NP
DEFINICION DE TABLAS DE VERDAD	: NP
NEGACION Y DOBLE NEGACION	: NP
CONJUNCION	: NP
DISYUNCION	: NP
IMPLICACION CONDICIONAL	: NP
DOBLE CONDICIONAL	: NP
TIPOS DE PROPOSICIONES	: NP
TAUTOLOGICAS	: NP
CONTRADICTORIAS	: NP
CONTINGENTES	: NP
VERDAD FORMAL Y VERDAD EMPIRICA	: NP

Figura 3.2.10 Pantalla de consulta para alumno o grupo (prototipo)

PAG. 3

CONSULTAR ALUMNO

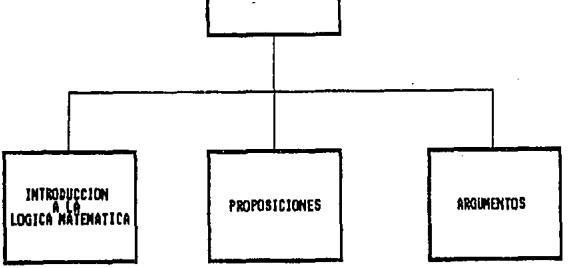
NOMBRE DE ALUMNO :
NO. DE LISTA :
NO. DE CUENTA :

LECCION 3.- ARGUMENTOS

COMPOSICION DE UN ARGUMENTO	: NP
VALIDEZ LOGICA DE UN ARGUMENTO	: NP
LEYES DE IMPLICACION	
MODUS PONENDO PONENS	: NP
MODUS TOLLENDO TOLLENS	: NP
MODUS TOLLENDO PONENS	: NP
SILOGISMO HIPOTETICO	: NP
LEY DE LA SIMPLIFICACION	: NP
LEY DE LA CONJUNCION	: NP
LEYES DE EQUIVALENCIA	: NP
DEMOSTRACION FORMAL DE LA VALIDEZ DE UN ARGUMENTO	: NP

El Subsistema de Instrucción incluye en su menú principal el nombre de las tres lecciones que conformaran dicho subsistema (figura 3.2.11) y de las cuales a su vez serán derivados los temas o subtemas inherentes respectivamente a cada lección.

Figura 3.2.11 Prototipo del Subsistema de Instrucción



A continuación se hace una breve descripción del contenido de cada una de las actividades generales de las partes componentes que estructuran al sistema tutorial LOGMAT.

3.2.1 Descripción del Subsistema de Control de Alumnos.

Las actividades generales de este subsistema son:

Altas	Registrar alumnos en la base de datos.
Bajas	Eliminar definitivamente los datos de un grupo de alumnos o de un alumno que hayan sido previamente registrados bajo el proceso altas.
Consultas	Consultar evaluaciones por grupo o por alumno, desplegando respectivamente la información en la pantalla.

3.2.2 Descripción del Subsistema de Instrucción.

Las actividades generales de este subsistema son:

- Registrar y validar que el alumno tenga acceso a utilizar el Subsistema de Instrucción.
- Enviar la orden de presentación de un tema o subtema proporcionando efectos especiales (movimiento, sonido, color, mensajes de estímulo, etc.).
- Enviar aleatoriamente la secuencia de ejercicios que el alumno deberá resolver.
- Recibir la respuesta del alumno para cada uno de los ejercicios.
- Interpretar la respuesta del alumno.
- Evaluar la respuesta del alumno comparándola con un patrón interno de respuestas correctas para posteriormente emitir la calificación.
- Almacenar en la base de datos la calificación que obtenga el alumno en algún tema o subtema.

3.2.3 Descripción de la base de datos.

La función de la base de datos es la de almacenar la información referente a los alumnos que interactuarán con el sistema tutorial LOGMAT manteniendo un control eficiente de la información de cada uno de ellos, así como el registro del aprovechamiento obtenido en los diversos temas o subtemas que conforman al Subsistema de Instrucción a efecto de poder brindarle al profesor una constante retroalimentación basada en información actualizada , oportuna y confiable.

La base de datos almacena la información de cada alumno tal como: nombre, número de lista, número de cuenta, así como la calificación de cada tema o subtema , es decir, la función que realiza la base de datos es la de gestionar la información que se obtenga de la ejecución de los procesos que realiza cada uno de los módulos del Subsistema de Control de Alumnos.

Interactuando con el prototipo funcional se lograron definir aquellas funciones que deberá realizar el sistema tutorial LOGMAT, ordenándolas en forma lógica, especificando las características de todos los datos que se deben registrar y las validaciones necesarias tanto para los datos como para los procesos.

3. 3 Definición de requerimientos finales.

Como ya se mencionó, la interacción del usuario (profesor y alumno) con el modelo funcional nos permitió definir al detalle aquella información considerada útil para realizar el sistema tutorial LOGMAT.

El objetivo, los requisitos operativos, metas y funciones, así como los requisitos funcionales a partir de los cuales se realizó el sistema tutorial LOGMAT son descritos a continuación:

3.4 Objetivo del sistema tutorial.

Servir como apoyo para la enseñanza de lógica matemática cumpliendo con la advertencia que hace el alumno de necesitar trabajar con materiales inasibles, manejar conceptos e ideas puras y aplicar teorías o razonamientos para colegir deducciones.

Requisitos operativos.

- El sistema tutorial debe de cumplir con el objetivo y requerimientos definidos.
- La utilización de los avances y las pruebas del sistema tutorial con datos reales, una vez definidos los objetivos y requerimientos de sistema, debe ser simultánea al desarrollo de los módulos de cada subsistema de tal manera que en la medida de lo posible sean coherentes con la asignatura de Matemáticas I dentro de los planes de estudio 1993.
- Debe ser sencillo, útil , confiable y con una interfase fácil de utilizar.
- El sistema tutorial debe motivar al alumno para que la utilización del mismo sea eficiente y productiva.
- El sistema tutorial debe ser compatible para funcionar en diversos equipos de computo.
- El sistema tutorial debe estar libre de defectos de diseño y codificación.
- Deben obtenerse resultados prácticos

3.5 Funciones de los procesos.

Se desglosan y se enlistan los funciones de cada uno de las procesos que son realizadas tanto dentro del Subsistema del Control de Alumnos así como en el Subsistema de Instrucción en base al orden que dentro del prototipo funcional se determinó y de acuerdo a las consideraciones funcionales y lógicas del mismo.

Funciones de los procesos del Subsistema de Control de Alumnos.

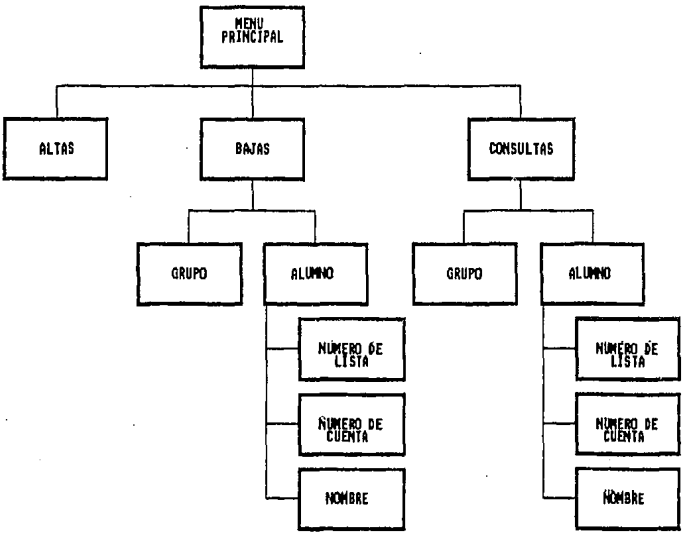
Las funciones de este subsistema son las siguientes a mencionar:

1. Altas.
   1.1 Alta por alumno.

2. Bajas.
   - 2.1 Baja por alumno.
     - 2.2.1 Baja por nombre.
     - 2.2.2 Baja por número de lista.
     - 2.2.3 Baja por número de cuenta.
3. Consultas.
   - 3.1 Consulta por grupo.
   - 3.2 Consulta por alumno.
     - 3.2.1 Consulta por nombre.
     - 3.2.2 Consulta por número de lista.
     - 3.2.3 Consulta por número de cuenta.

Se puede observar la estructuración modular de los procesos y las funciones del Subsistema de Control de Alumnos en el figura 3.5.1.

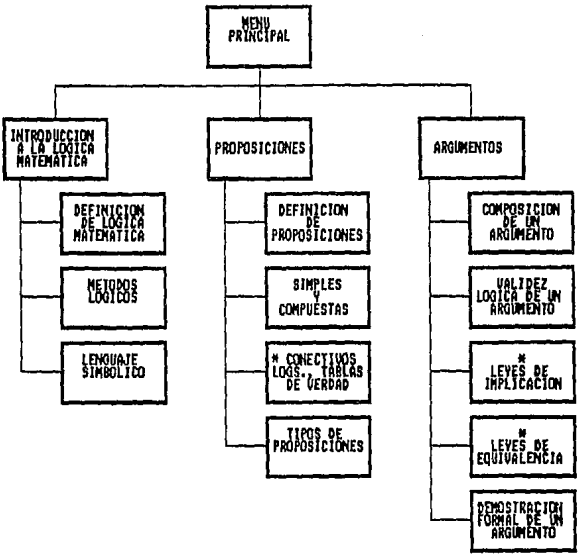
Figura 3.5,1 Subsistema de Control de Alumnos



Función del proceso del Subsistema de Instrucción.

De acuerdo con el criterio de la ingeniería de software en la estructuración del contenido temático del Subsistema de Instrucción se determinó que su funcionamiento se haría de forma modular con el propósito de facilitar su manejo y operación. La estructuración del contenido temático del Subsistema de Instrucción se muestra en el figura 3.5.2.

Figura 3.5.2 Subsistema de Instrucción



* Existen subtemas asociados a estos temas.

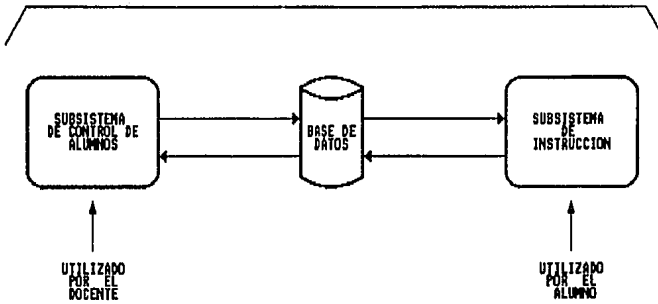
Donde el Subsistema de Instrucción está compuesto de tres lecciones principales. A su vez cada lección se divide en un conjunto finito de temas y/o subtemas.

Alcance del sistema tutorial.

En la sección anterior quedaron explicados el objetivo, los requisitos operativos, las metas y las funciones del sistema; estos representan los requerimientos finales autorizados, en base a los cuales se desarrolló el sistema tutorial.

En la figura 3.5.3 se delimita el alcance del sistema tutorial, el origen de la información y su destino final.

Figura 3.5.3 Alcance del sistema tutorial LOGMAT.



3.6 Metodología para la realización de sistema tutorial.

La metodología para el desarrollo del software que fue utilizada durante la realización del Subsistema de Instrucción que conforma al sistema tutorial LOGMAT se basa en los conceptos y principios de Alfred Bork (capítulo 2) en sus siguientes etapas:

Desarrollo de la etapa del diseño pedagógico inicial.

¿Qué es lo que queremos que el programa haga?, ¿a quién va destinado? En este momento son definidos los objetivos educativos que se pretenden, alumnos a los que irá destinado, se contemplan las lecciones, temas y subtemas a tratarse (capítulo 5, tabla 5.4.1 lecciones, temas y subtemas abordables en el sistema tutorial) basándose para ello de consultar los planes analíticos de estudio 1985 y 1993 de la FCA además de ser también consultados libros sobre lógica matemática.

Desarrollo de la etapa del diseño pedagógico detallado.

Durante esta etapa se trabaja en cada tema o subtema a ser empleado dentro del Subsistema de Instrucción con la finalidad de producir el guión para cada módulo de aprendizaje tomándose en cuenta los contenidos temáticos que se tratarán (hechos, conceptos, leyes, procedimientos, valores, etc.), también se analizan las repuestas del alumno y se definen las decisiones que deberán tomarse en base a este análisis sin olvidar el caso de la respuesta no esperada.

Desarrollo de la etapa del diseño de pantalla.

En esta etapa es diseñada la manera en que el material aparecerá en la pantalla, teniendo en cuenta la posición espacial que deberán tener tanto los textos (cuidando de que sean presentados la mínima cantidad de ellos por cada pantalla) como los figuras y sin descuidar omitir el tiempo de aparición del material en la pantalla. En los figuras 3.6.1 y 3.6.2 se observa el modelo de guión que se utilizó dentro del Subsistema de Instrucción para diseñar la forma de codificación a ser desarrollada en las pantallas de presentación de los contenidos temáticos y ejercicios respectivamente.

Figura 3.6.1 Guión para presentación del contenido temático

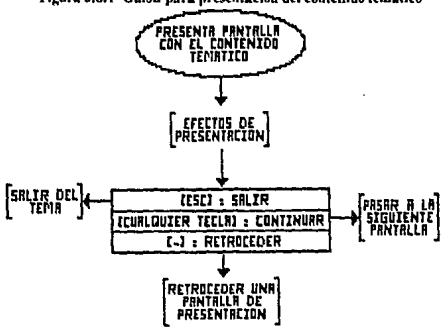
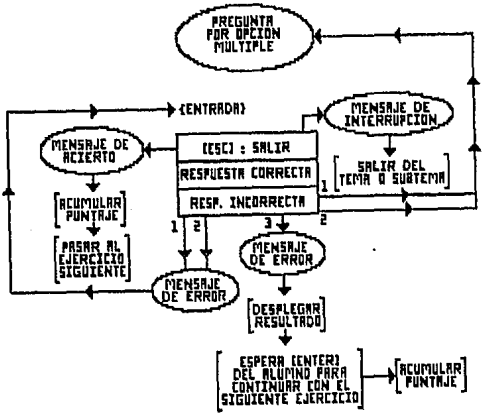


Figura 3.6.2 Guión para presentación del ejercicio



Desarrollo de la etapa de codificación.

Son preparadas las pantallas gráficas e historias temáticas haciendo un plan paralelo del código bajo una estructura modular que permita su optimización y futuro mantenimiento (figuras 3.6.3 a 3.6.12); en esta etapa se lleva a cabo la codificación del Subsistema de Instrucción.

Figura 3.6.3 Pantalla del sistema tutorial LOGMAT perteneciente a la lección 1 Introducción a la Lógica Matemática, tema 1.1 Definición de Lógica Matemática.

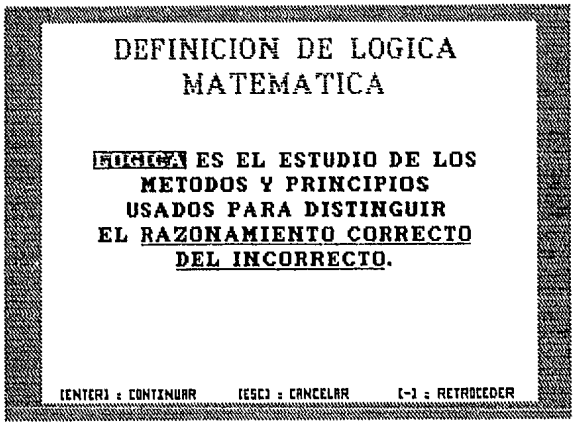


Figura 3.6.4 Pantalla del sistema tutorial LOGMAT perteneciente a la lección 2 Proposiciones, tema 2.3 Conectivos Lógicos y Tablas de Verdad, subtema 2.3.5 Disyunción.

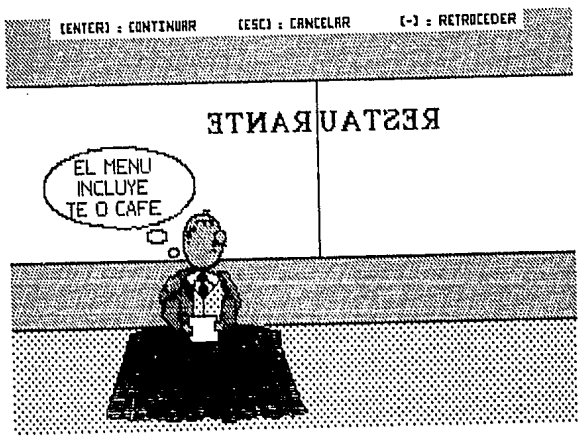


Figura 3.6.5 Pantalla del sistema tutorial LOGMAT perteneciente a la lección 3 Argumentos, tema 3.1 Composición de un Argumento.

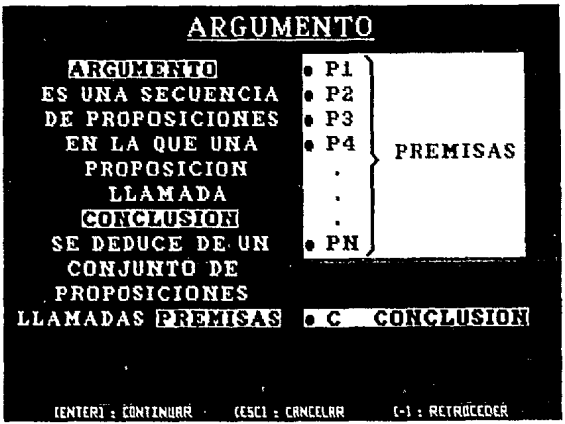


Figura 3.6.6 Pantalla del sistema tutorial LOGMAT perteneciente a la lección 3 Argumentos, tema 3.3 Leyes de Implicación, del subtema 3.3.6 Ley de la Conjunción.

Figura 3.6.7 Pantalla del sistema tutorial LOGMAT perteneciente a la lección 3 Argumentos, tema 3.5 Demostración Formal de la Validez Lógica de un Argumento.

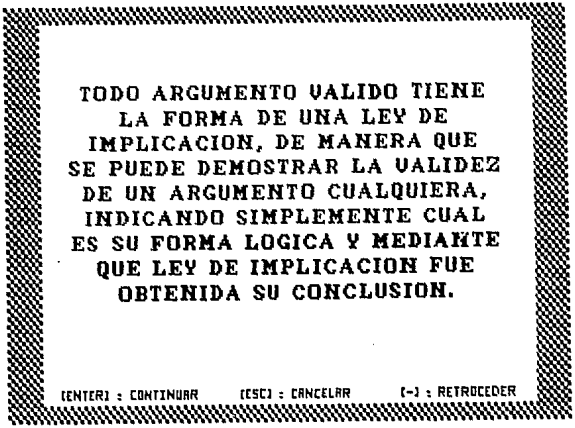


Figura 3.6.8 Pantalla de ejercicio del sistema tutorial LOGMAT perteneciente a la lección 1 Introducción a la Lógica Matemática, tema 1.1 Definición de Lógica Matemática.

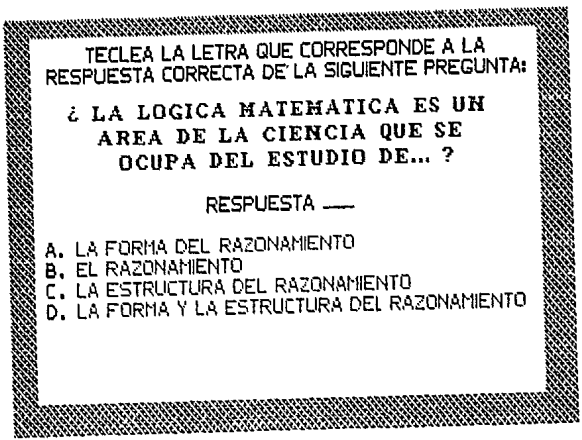


Figura 3.6.9 **Pantalla de ejercicio del sistema tutorial LOGMAT perteneciente a la lección 2 Proposiciones, tema 2.1 Definición de Proposiciones.**

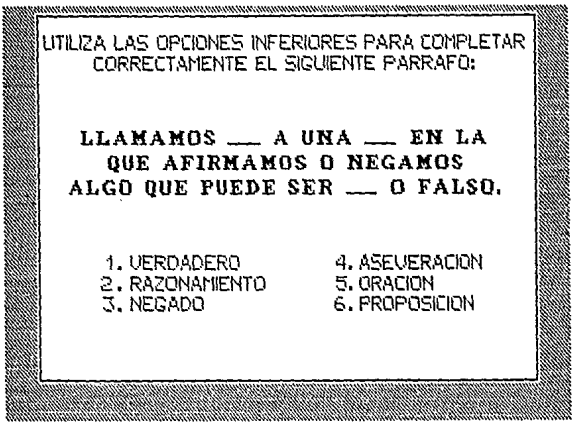


Figura 3.6.10 **Pantalla de ejercicio del sistema tutorial LOGMAT perteneciente a la lección 2 Proposiciones, tema 2.3 Conectivos Lógicos y Tablas de Verdad, subtema 2.3.3 Negación y Doble Negación.**

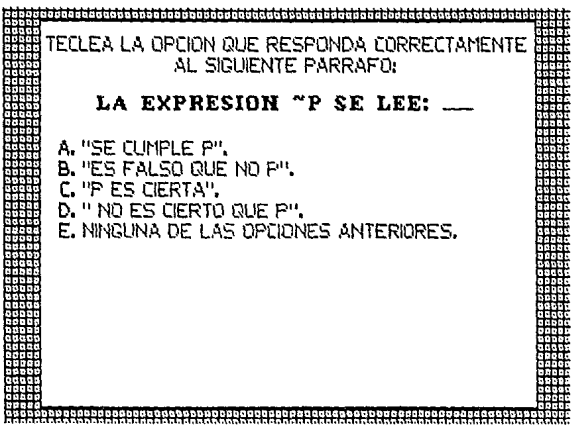


Figura 3.6.11 Pantalla de ejercicio del sistema tutorial LOGMAT perteneciente a la lección 2 Proposiciones, tema 2.3 Conectivos Lógicos y Tablas de Verdad, subtema 2.3.7 Doble Condicional.

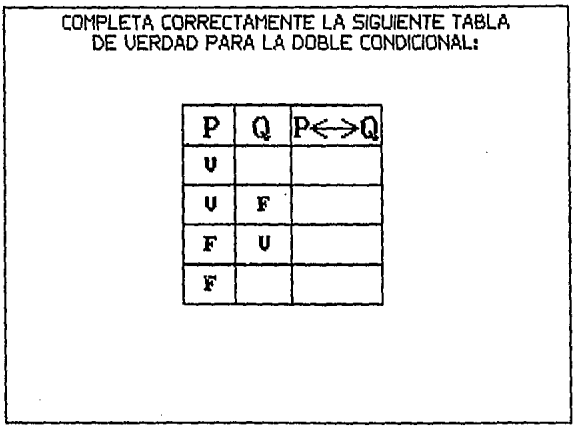
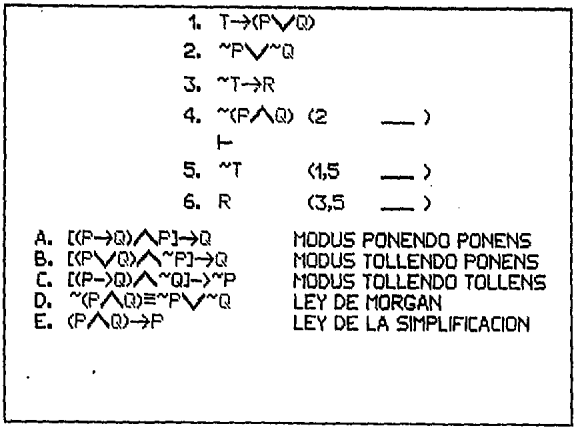


Figura 3.6.12 Pantalla de ejercicio del sistema tutorial LOGMAT perteneciente a la lección 3 Argumentos, tema 3.5 Demostración Formal de la Validez de un Argumento.



3.7 Arquitectura de software.

La arquitectura de software debe contemplar dos aspectos: 1) la jerarquía de sus módulos, y 2) la estructura de los datos. De la jerarquía de los módulos, se obtiene la división del problema en funciones que resuelven una acción en específico. Esta división del problema se realiza durante la definición de requerimientos con la ayuda de los diagramas de flujo de datos y debe representarse en forma gráfica para que sirva de guía en la codificación y forme parte del sistema tutorial cuando esté terminado.

Un diagrama jerárquico es una herramienta práctica que permite ilustrar la estructura jerárquica de los subsistemas especificando la secuencia y relación entre cada uno de los módulos, lo cual lo hace sumamente útil para representar el sistema a desarrollar.

La jerarquía del software del sistema tutorial se obtiene del análisis de las funciones realizadas con el prototipo funcional lo cual permite definir y validar la estructura, logrando que ésta fuera funcional y que cada módulo tuviera asignada la función de un aspecto específico.

La disquisición de cada módulo se vincula en la descripción del diagrama jerárquico del Subsistema de Control de Alumnos y del Subsistema de Instrucción.

3.8 Diagramas de proceso por módulo.

El diseño procedimental transfiere cada módulo a una descripción a detalle del proceso que realizará; es decir, describe los detalles de proceso necesarios en cada módulo, estableciendo la lógica, las decisiones de proceso y la secuencia de cada actividad.

El mencionado diseño transforma los elementos estructurales en una descripción procedimental de software y se realiza después de que se ha establecido la estructura del programa y los datos. Es en la especificación de cada proceso donde se definen los detalles algoritmicos que producen la generación del código fuente y conllevan a la integración y validación del software.

En base a la definición de requerimientos y al diseño arquitectónico (3.1 y 3.7, de este capítulo) presentamos a continuación, los aspectos importantes que se deben considerar durante la construcción de los programas. Estas consideraciones se contemplan mediante diagramas de proceso en los que se hace la descripción del flujo de proceso algorítmico más relevante para efectuar la codificación (diagramas 3.8.1, 3.8.1-A, 3.8.1-B, 3.8.1-C y 3.8.2).

Figura 3.8.1 Subsistema de Control de Alumnos
Diagrama de proceso general

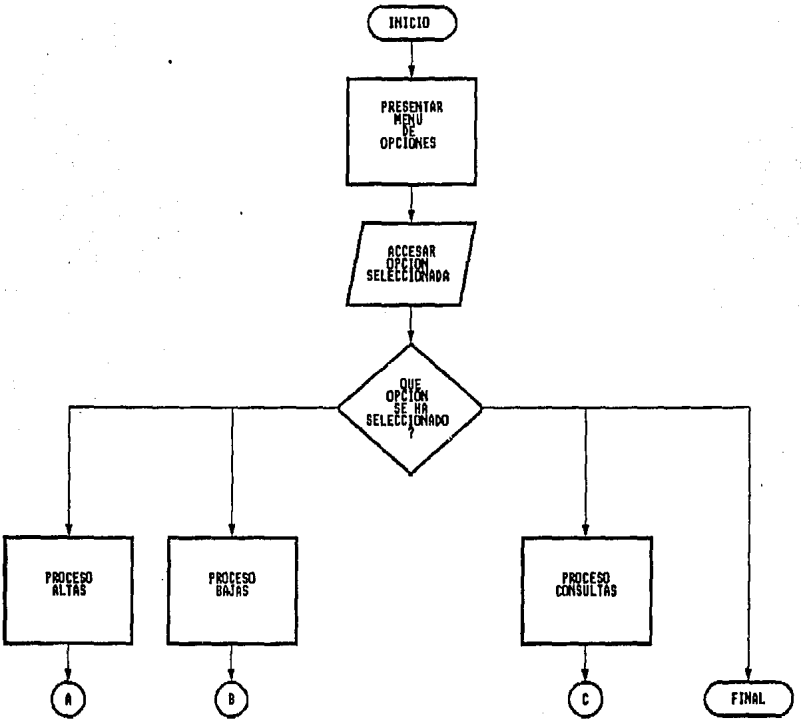


Figura 3.8.1-A Subsistema de Control de Alumnos.
Diagrama de Proceso (Altas)

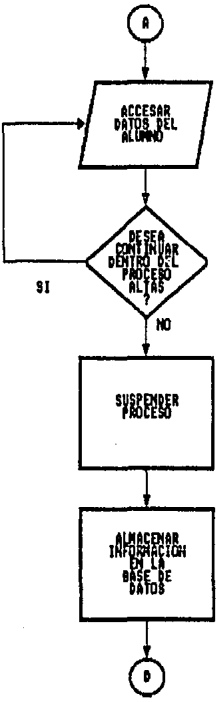


Figura 3.8.1-B Subsistema de Control de Alumnos.
Diagrama de Proceso (Bajas)

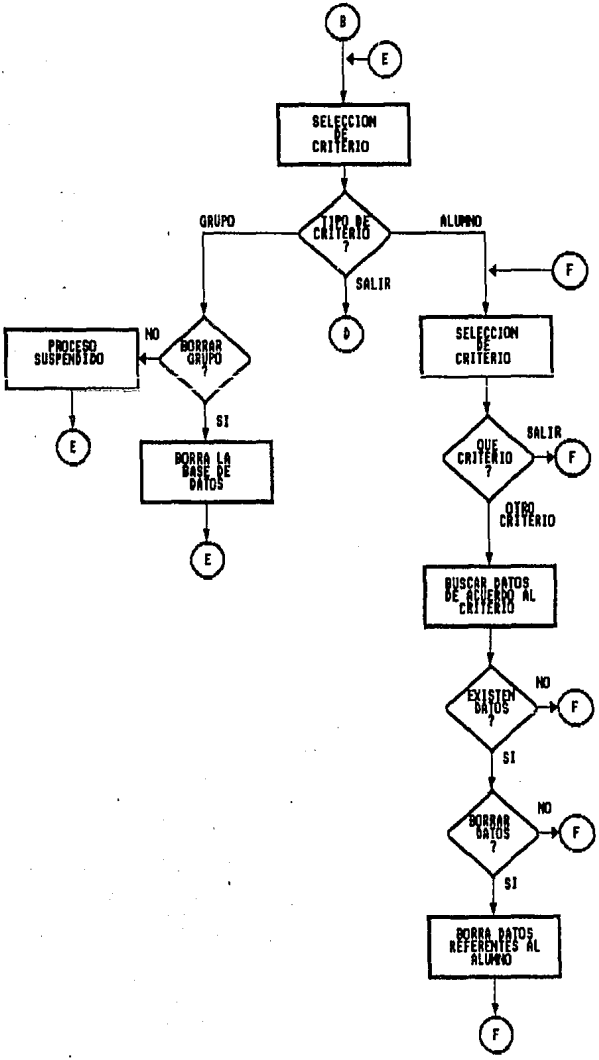


Figura 3.8.1-C Subsistema de Control de Alumnos.
Diagrama de Proceso (Consultas)

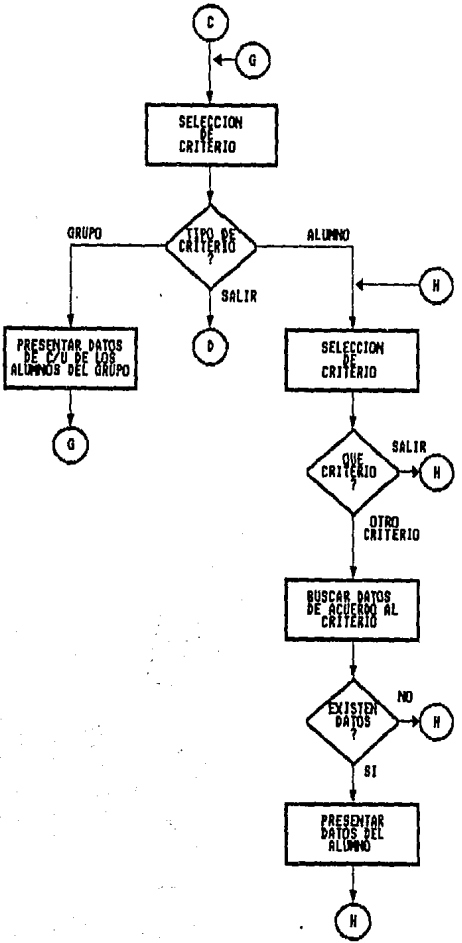
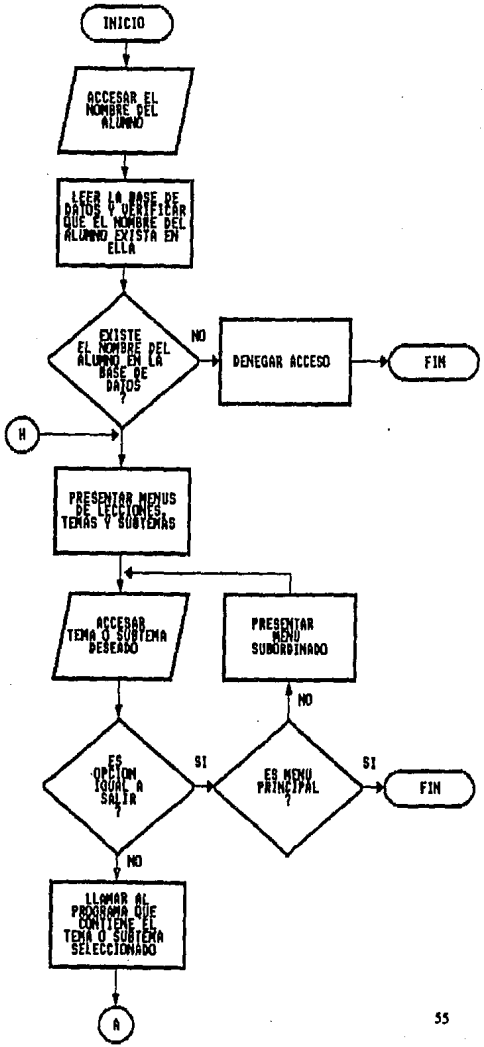
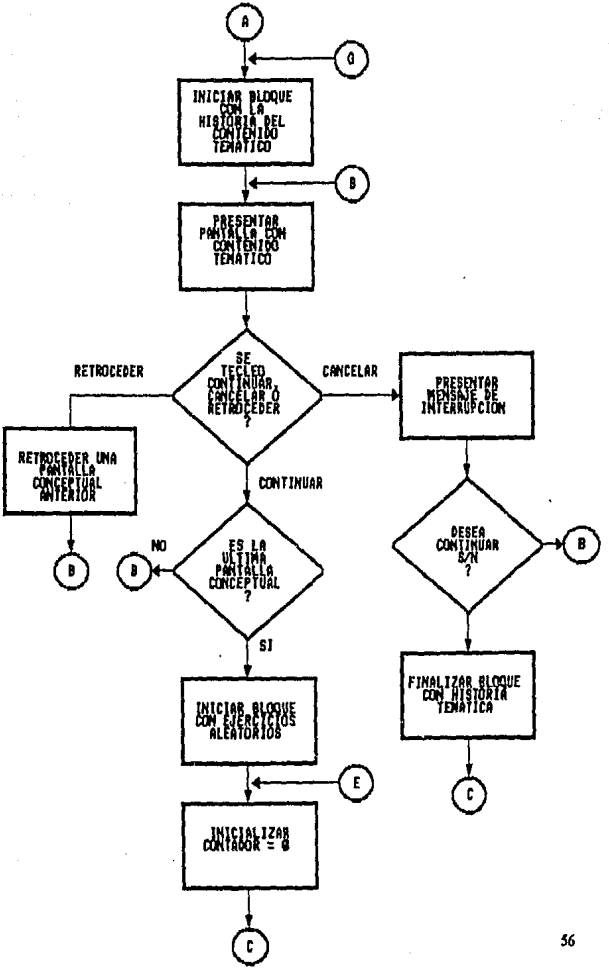


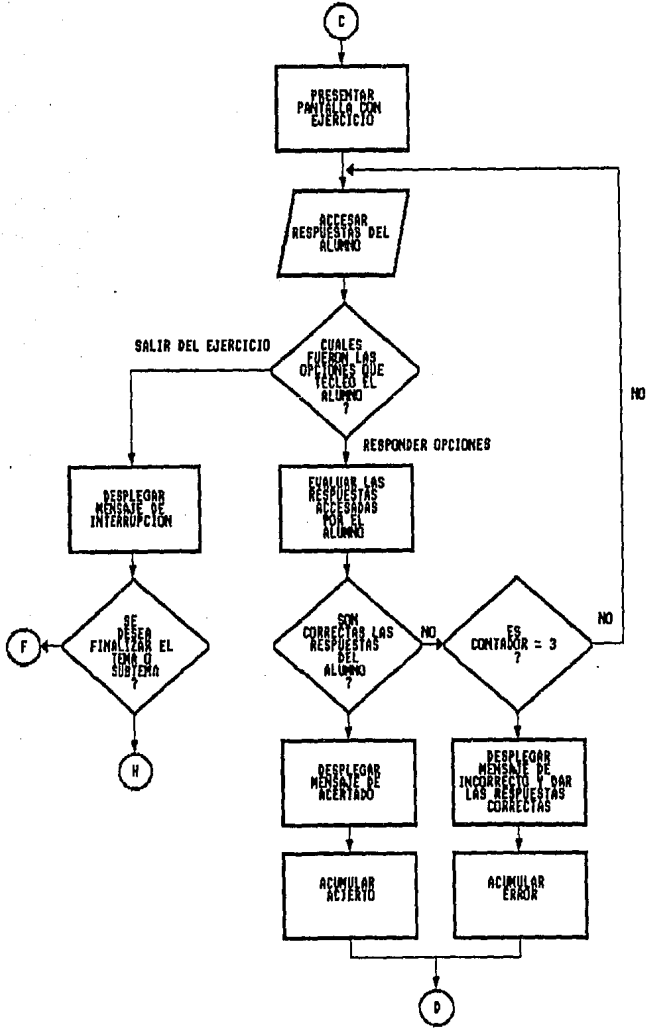
Figura 3.8.2 Subsistema de Instrucción.
Diagrama de Proceso General



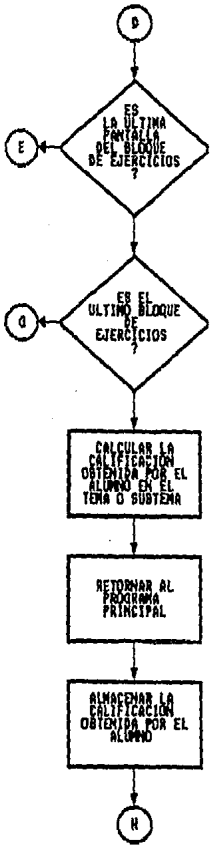
Subsistema de Instrucción.
Diagrama de Proceso General (continuación)



Subsistema de Instrucción.
Diagrama de proceso General (continuación)



Subsistema de Instrucción.
Diagrama de proceso General (continuación)



3.9 Estructura de la base de datos.

El diseño estructural de la base de datos se llevó a cabo empleando la técnica de lista doblemente enlazada ya que esta permite ser utilizada para dos propósitos principales.

- Crear en memoria un arreglo que tenga longitud desconocida.
- Proporcionar el almacenamiento de la base de datos en un archivo permitiendo insertar y borrar elementos de manera fácil y rápida sin tener que reordenar el archivo en el disco.

La lista doblemente enlazada provee de un enlace tanto al elemento siguiente como al anterior. La aplicación de este tipo de lista con doble unión tiene dos grandes ventajas. La primera de ellas es que la lista se puede leer hacia atrás o hacia adelante lo cual no sólo simplifica la ordenación de la lista, sino que también le permite al usuario recorrer la lista en las dos direcciones. La segunda consiste en que dado que la lista entera se puede leer en dos direcciones, al ser invalidado alguno de ellos (significativo en el caso de fallo del equipo) la lista puede volver a ser reconstruida utilizando el otro enlace.

El siguiente paso para diseñar la estructura de la base de datos fue distinguir dentro de los datos requeridos las diferentes entidades sobre las que se almacenarán datos de identificación del alumno y de evaluación. Esta consideración condujo a utilizar una base de datos para el manejo de los datos relacionados con los alumnos. Esta conceptualización estructural permite almacenar información con las características que se describen en la figura 3.9.1.

Figura 3.9.1 Tabla descriptiva de la base de datos

Campo	Tipo	Longitud	Descripción
NOLISTA	Caracter	02	Número de lista.
NOCUENTA	Caracter	09	Número de cuenta.
NOMBRE	Caracter	10	Nombre.
CLDEFLOG	Caracter	03	Definición de lógica matemática.
CLLENSIM	Caracter	03	Lenguaje simbólico.
CLMETLOG	Caracter	03	Métodos lógicos.
CLDEFPRO	Caracter	03	Definición de proposiciones.
CLSIMCOM	Caracter	03	Simples y compuestas
CLDEFCON	Caracter	03	Definición de conectivos lógicos.
CLDEFTAV	Caracter	03	Definición de tablas de verdad.
CLNEGDNE	Caracter	03	Negación y doble negación.
CLCONJUN	Caracter	03	Conjunción.
CLDISYUN	Caracter	03	Disyunción.
CLIMPCON	Caracter	03	Implicación condicional.
CLDOBCON	Caracter	03	Doble condicional.
CLTAUTOL	Caracter	03	Tautológicas.
CLCONTRA	Caracter	03	Contradictorias.
CLCONTIN	Caracter	03	Contingentes.
CLVRMVEN	Caracter	03	Verdad formal y verdad empírica.
CLCOMARG	Caracter	03	Composición de un argumento.
CLVALOAR	Caracter	03	Validez lógica de un argumento
CLMOPOPO	Caracter	03	Modus ponendo ponens
CLMOTOTO	Caracter	03	Modus tollendo tollens
CLMOTOPO	Caracter	03	Modus tollendo ponens
CLSILHIP	Caracter	03	Silogismo hipotético
CLLEYSIM	Caracter	03	Ley de la simplificación
CLLEYCON	Caracter	03	Ley de la conjunción
CLLEYADI	Caracter	03	Ley de la adición
CLLEYEQU	Caracter	03	Leyes de equivalencia
CLDEFVAR	Caracter	03	Definición formal de la validez de un argumento

4. CODIFICACION E IMPLEMENTACION.

4.1 Metodología e implementación.

Para la construcción del Subsistema de Instrucción del sistema tutorial LOGMAT se empleo la metodología propuesta por A. Bork, en sus etapas de diseño pedagógico, diseño de pantalla, codificación, análisis, evaluación y revisión (capítulo 3, 3.6).

Implementación física.

Dentro de la implementación del sistema tutorial se tomaron en cuenta las características de instrumentación disponibles en diferentes lenguajes, seleccionando el lenguaje que permitiera construir y administrar eficientemente la base de datos y los procesos del sistema tutorial a desarrollar y considerando además el funcionamiento del sistema para microcomputadoras.

Con fundamento en lo anterior en la fase de implementación física se utilizó:

El lenguaje de programación Turbo C 2.0 (de Borland International, 1988) que tiene como característica principal ser un lenguaje de alto nivel de aplicación general el cual permite la creación de gráficos y efectos especiales, además de poseer un moderno flujo de control y estructura de datos.

El paquete Story Board Plus 2.0 (marca registrada de IBM, 1989) para la creación de la secuencia temática a través de historias computarizadas.

4.2 Estilo y estándar de codificación.

La adopción del estilo de codificación representa la forma específica que se selecciona para llevar a cabo la realización de un sistema cuyo propósito se destinó a crear código comprensible y sencillo.

La adopción del estilo y los estándares de codificación que se describen en este capítulo manifiestan el estilo personal de la autora. Se trata de un estilo categórico, pero inteligible.

La codificación del sistema tutorial esta basada en la utilización de criterios de modularidad que Roger S. Pressman [1] señala dentro de la construcción del diseño modular de todo programa:

Cohesión: Es la manera de medir si existe fuerza funcional dado un procedimiento siempre y cuando dicho procedimiento realice una sola función. Un módulo cohesivo ejecuta una tarea sencilla de un procedimiento de software y requiere poca interacción con procedimientos que ejecutan otras partes de un programa, es decir, un módulo cohesivo sólo hace (idealmente) una cosa.

Acoplamiento: Es la manera de medir la dependencia de interconexión entre los módulos. Preferentemente se busca que durante el diseño se tenga un acoplamiento bajo, con el objeto de lograr una mejor comprensión; evitando así que los errores de un módulo se extiendan a otros módulos durante la ejecución del sistema.

---

[1] Pressman R. S. "Ingeniería de Software: un enfoque práctico", (3a. ed), México, McGraw-Hill, 1993, pp. 349-354.

En base a estos criterios fue estructurado el sistema tutorial, estableciendo que cada tema o subtema dentro del Subsistema de Instrucción fuera desarrollado en un programa específico. Este programa se interrelaciona con el programa principal del subsistema mediante parámetros auxiliado con rutinas globales almacenadas en librerías, a fin de evitar la redundancia en el código.

En la figura 4.2.1 se observa que la descripción de la estructura de cada módulo o función del Subsistema de Instrucción corresponden a un solo programa, lo cual permite lograr que el acoplamiento que existe entre cada programa sea bajo. Este acoplamiento únicamente existe con respecto al contenido ya que, si bien, no se manifiesta un acoplamiento directo (todos los programas se encuentran estrechamente relacionados) si se manifiesta que todos los programas accesen a la misma base de datos que el Subsistema de Control de Alumnos genera.

Figura 4.2.1 Descripción de la estructura del Subsistema de Instrucción

NOMBRE DEL MODULO O FUNCION	DESCRIPCION DEL MODULO O FUNCION	NOMBRE DE PROGRAMAS O FUNCIONES	NOMBRE Y RANGO DE HISTORIAS TEMATICAS	NOMBRE Y RANGO DE LOS GRAFICOS
Menú Principal	Representa el menú a partir del cual se podrá elegir la lección, el tema o el subtema deseado	LOOMAT	UNAM-FCA.SH~, INICIO.SH~	MENU1.PIC a MENU8.PIC y LUMNO.PIC.
Tema 1.1	Definición de Lógica Matemática	SIDEFLOG	LOGICAP1.SH~	LITIE1.PIC a LITIE6.PIC.
Tema 1.2	Métodos Lógicos	SIMETLOG	METODO1.SH~ a METODO4.SH~.	LITZE1.PIC a LITZE6.PIC.
Tema 1.3	Lenguaje Simbólico	SILENSIM	LENSIM1.SH~ a LENSIM5.SH~.	LITSE1.PIC a LITSE12.PIC.
Tema 2.1	Definición de Proposiciones	SIDEFPRO	PROPO1.SH~ a PROPO4.SH~.	LT2IE1.PIC a LT2IE16.PIC.
Tema 2.2	Simples y Compuestas	SISIMCOM	SIMCOMPU.SH~	LT2SE1.PIC a LT2SE6.PIC.
Subtema 2.3.1	Definición de Conectivos Lógicos	SIDEFCON	CONECT1.SH~ a CONECT2.SH~.	LT3S2E1.PIC a LT3S2E6.PIC.
Subtema 2.3.2	Definición de Tablas de Verdad	SIDEFTAV	TABVER1.SH~ a TABVER3.SH~.	LT3S2E1.PIC a LT3S1ED.PIC.
Subtema 2.3.3	Negación y Doble Negación	SINEGDNE, NEG y NEGD	NEGST1.SH~ a NEGST2.SH~	LT3S4E1.PIC a LT3S4EK.PIC, TABLANEG.PIC y TABLNEGD.PIC
Subtema 2.3.4	Conjunción	SICONJUN y CONJUNC1	AND1.SH~ a AND2.SH~.	LT3S4E1.PIC a LT3S4EJ.PIC, TABCONJUN.PIC.
Subtema 2.3.5	Disyunción	SIDISYUN, DISY1 y DISY8	DISYUN1.SH~ a DISYUN2.SH~.	LT3S5E1.PIC a LT3S3EP.PIC, TABDISY1.PIC y TABDISY8.PIC.
Subtema 2.3.6	Implicación Condicional	SIIMPCON y IMPCON	IMPCON1.SH~ a IMPCON4.SH~.	LT3S6E1.PIC a LT3S6EF.PIC, TABIMPCON.PIC
Subtema 2.3.7	Doble Condicional.	SIDOBCON y DOBCON	DOBCON1.SH~ a DOBCON4.SH~.	LT3S7E1.PIC a LT3S7ED.PIC, TADOBCON.PIC
Subtema 2.4.1	Tautológicas	SITAUTOL	TAUTOL01.SH~ a TAUTOL03.SH~.	LT4S1E1.PIC a LT4S1ED.PIC.
Subtema 2.4.2	Contradictorias	SICONTRA	CONTRA1.SH~ a CONTRA3.SH~.	LT4S2E1.PIC a LT4S2ED.PIC.
Subtema 2.4.5	Contingentes	SICONTIN	CONTIG1.SH~ a CONTIG3.SH~.	LT4S3IE1.PIC a LT4S3IED.PIC.
Tema 2.5	Verdad Formal y Verdad Empírica.	SIVEREM	VERFOEMP.SH~	LT2SH1.PIC a LT2SH4.PIC.
Tema 3.1	Composición de un Argumento	SICOMARG	ARGUMEN1.SH~ a ARGUMEN3.SH~.	LT3E2E1.PIC a LT3E2E9.PIC.
Tema 3.2	Validez Lógica de un Argumento.	SIVALOAR	VALARG1.SH~ a VALARG10.SH~.	LT3S3IE1.PIC a LT3S3E26.PIC.
Subtema 3.3.1	Modus Ponendo Ponens	SIMOPO	MPP1.SH~ a MPP3.SH~.	LT3S3IE1.PIC a LT3S3IED.PIC.
Subtema 3.3.2	Modus Tollendo Tollens	SIMOTO	MTT1.SH~ a MTT3.SH~.	LT3S3E1.PIC a LT3S3ED.PIC.
Subtema 3.3.3	Modus Tollendo Ponens.	SIMOPO	MTP1.SH~ a MTP3.SH~.	LT3S3E1.PIC a LT3S3ED.PIC.
Subtema 3.3.4	Silogismo Hipotético.	SISILHIP	SH1.SH~ a SH2.SH~.	LT3S4E1.PIC a LT3S4ED.PIC.
Subtema 3.3.5	Ley de la Simplificación.	SILEYSIM	SIM1.SH~ a SIM2.SH~.	LT3S5E1.PIC a LT3S5ED.PIC.
Subtema 3.3.6	Ley de la Conjunción.	SILEYCON	LCONSTOR.SH~	LT3S6E1.PIC a LT3S6ED.PIC.
Subtema 3.3.7	Ley de la Adición	SILEYADI	ADDSTORY.SH~	LT3S7E1.PIC a LT3S7ED.PIC.
Tema 3.4	Leyes de Equivalencia	SILEYEQU	EQUI1.SH~ EQUI2.SH~.	LT4E1.PIC a LT4E30.PIC.
Tema 3.5	Demostración Formal de la Validez de un Argumento	SIDEMVAR	DEVARG5T.SH~.	LT5E1.PIC a LT5E14.PIC.

Por lo que a la cohesión respecta es alta en cada módulo del Subsistema de Instrucción ya que cada programa está formado de varias rutinas que en su conjunto realizan una función de validación de ejercicios específicos lo cual proporciona que cada rutina represente una unidad funcional coherente realizando una sola cosa y logrando como resultado la cohesión de todos los programas que conforman el sistema tutorial.

Con el propósito de mantener la consistencia del estilo se buscó realizar una codificación fuente (ejemplo de código fuente, Anexo A) apropiada, basándose en los siguientes criterios:

- Para cada programa incluir un encabezado con la siguiente información:

  Nombre del programa.
  Nombre del sistema.
  Nombre del autor.
  Fecha de la última modificación.
  Descripción breve del objetivo del programa.
  Programa llamado por...
  Programa que llama a...
  Parámetros que recibe.

- Utilizar dentro de los procesos similares las mismas proposiciones de control de programas (IF-ELSE, SWITCH, FOR, WHILE, DO-WHILE).

- Codificar proposiciones lógicas IF-ELSE de manera que esto evite conducirnos a tener proposiciones nulas.

- Evitar redundar en código innecesario si este se puede crear en una rutina global e incluirse como directiva del preprocesador.

- Cuidar que las construcciones anidadas se encuentren indentadas apropiadamente con el objeto de evitar niveles muy profundos de anidamiento.

- Buscar métodos alternativos que eviten las existencia de rutinas complejas.

La elección de un estilo en sí, genera un estándar para codificar programas permitiendo así lograr homogeneidad.

Una manera adicional de mantener un estándar es provista por aspectos relacionados con la identificación de: los programas, las bases de datos, los campos de la base de datos así como los nombres de las variables de control y captura que son empleados en los programas.

Estos aspectos tienen como propósito perfeccionar la transparencia del código, controlar eficientemente el flujo de los datos, facilitar la identificación de los nombres (programas, bases de datos, variables, parámetros, etcétera). Con ello se evita crear innecesariamente variables que consuman memoria o que lleven a originar inconsistencia en la información además de que es posible determinar con exactitud el contenido de las variables de control y de captura.

Programas.

En la identificación del programa que gestiona al Subsistema de Control de Alumnos se utilizó como identificación el nombre ALUMNOS.C.

En la identificación del programa principal del Subsistema de Instrucción se emplearon las tres primeras letras de lógica matemática, este programa tiene por nombre LOGMAT.C, en la identificación de los programas que hacen referencia a los temas y subtemas de dicho subsistema se empleó un prefijo de dos letras (SI) y seis para identificar el tema o subtema. Por ejemplo, el programa cuyo contenido temático es la "VALIDEZ LÓGICA DE UN ARGUMENTO" se llama SIVALOAR.C; el programa que hace referencia al contenido temático de la "CONJUNCIÓN" se llama SICONJUN.C y así sucesivamente.

Base de datos.

Para identificar a la base de datos se utilizó el nombre "ALUMNOS" y un prefijo de tres caracteres adicionales (DAT) que representan la extensión, la base de datos se identifica por el nombre de ALUMNOS.DAT.

Campos en la base de datos.

Para identificar a los campos se utilizó un prefijo de dos letras CL (calificación) y seis caracteres que identifican el tema o el subtema referente. Por ejemplo, el campo que hace referencia a la calificación del tema "DEFINICIÓN DE LÓGICA MATEMÁTICA" se llama CLDEFLOG.

Historias temáticas y gráficos.

En la identificación de las historias temáticas y de los gráficos relacionados a ellas se utilizan nombres mnemónicos sin una construcción especial pero utilizados en forma consistente y empleándose tanto en las historias como en los gráficos las extensiones que provee Story Board, (SH~) y (PIC) respectivamente.

Para identificar los gráficos de los ejercicios realizados también en Story Board con extensión (PIC) se utilizó una combinación de letras y números. Por ejemplo para identificar el décimo ejercicio del tema 2.1 DEFINICIÓN DE PROPOSICIONES se llama L2T1E10.PIC donde (L2) hace referencia a la lección 2, (T1) al tema 1 de dicha lección y (E10) al décimo ejercicio considerándose el empleo de los números (0...9) para los primeros diez ejercicios y letras (A...Z) para la identificación de aquellos ejercicios que excedan de más de diez, el onceavo ejercicio del subtema 2.3.5 DISYUNCIÓN se llama L2T3S5EA.PIC donde (L2) hace referencia a la lección 2, (T3) al tema 3, (S5) al subtema 5 de dicha lección y (EA) al onceavo ejercicio.

Variables de control y variables de captura.

Para la identificación de estas variables se utilizan nombres mnemónicos sin una construcción especial pero utilizados de manera consistente.

4.3 Configuración del hardware.

Es importante definir inicialmente la configuración del hardware en la que deberá operar un sistema, ya que con ella se medirá la velocidad de procesamiento y la eficiencia de la interacción con el usuario. A continuación son presentadas las propuestas de dos tipos de configuraciones:

Configuración mínima.

Requerida para el funcionamiento del sistema tutorial LOGMAT:

- Procesador 8088.
- Monitor monocromático CGA.
- Disco duro de 20 MB.
- Velocidad de procesamiento de 8 MHz.
- Unidad de disco flexible de baja densidad de 5¼ o 3½.
- Memoria RAM de 640 KB.
- Sistema operativo MS-DOS versión 3.0.

Configuración máxima.

Recomendable para un mejor funcionamiento del sistema tutorial LOGMAT:

- Procesador 80386/80486.
- Monitor policromático SUPER VGA.
- Disco duro de 100 MB.
- Velocidad de procesamiento de 25 MHz.
- Unidad de disco flexible de alta densidad de 5¼ o 3½.
- Memoria RAM de 640 KB.
- Memoria Extendida de 4.0 MB.
- Sistema operativo MS-DOS versión 5.0 o posteriores.

Cabe señalar que pueden emplearse configuraciones intermedias; no obstante, las pruebas de ejecución del sistema tutorial sólo se realizaron para las configuraciones anteriormente expuestas.

5. RESULTADOS.

5.1 Informática educativa: un área de compromiso.

Este trabajo intenta mostrar que las computadoras pueden resolver algunos de los problemas educativos, en particular problemas de índole cualitativa. Asimismo, se insiste en que no basta con el solo hecho de que la computación moderna ofrezca tal capacidad, sino que para sacar provecho educativo a una computadora se requieren acciones concertadas entre miembros de los dos sectores -educación e informática- que permitan maximizar el uso de este recurso. Adicionalmente los esfuerzos en informática educativa para ser valiosos en ambos sectores, deberán atender necesidades educativas prioritarias antes que centrarse en el desarrollo de la Informática Educativa como un fin en sí misma.

5.2 Proceso de producción.

En esta segunda parte de los resultados se reporta la descripción de la producción de material de aprendizaje basado en computadoras empleando la metodología que A. Bork propone dentro del desarrollo de sus etapas (capítulo 3).

5.3 Interfase modular del sistema tutorial.

La interfase modular del sistema tutorial LOGMAT se establece mediante un flujo de información y control el cual tiene como funciones las dos siguientes a mencionar:

- Los subsistemas interactuantes que conforman el sistema tutorial.
- La definición del flujo de información entre esos subsistemas.

Esta interfase ha sido construida a partir de la identificación de dos subsistemas: Subsistema de Control de Alumnos y Subsistema de Instrucción.

Identificación del Subsistema de Control de Alumnos.

El propósito de este subsistema es el de proporcionar al profesor una herramienta que le permita realizar las siguientes tareas a mencionar:

- Dar de alta en la base de datos a los alumnos que conformarán el grupo de trabajo y utilizarán el sistema tutorial.

- Eliminar a un alumno o a un grupo de alumnos de la base de datos cuando el profesor así lo requiera.

- Consultar la base de datos para proporcionar el avance y la evaluación de todos o de cada uno de los alumnos que han utilizado el sistema tutorial.

Otra función deseable en el Subsistema de Control de Alumnos es:

- Proporcionar la administración de la información de cada alumno. Función recomendada cuando el sistema tutorial es utilizado por un grupo de alumnos.

Identificación del Subsistema de Instrucción.

El propósito de este subsistema es el de interactuar con el alumno bajo el ambiente que el subsistema provee:

- Decidir por el alumno el despliegue en pantalla del tema o subtema deseado.
- Recuperar las historias temáticas almacenadas en los archivos gráficos.
- Recuperar aleatoriamente el bloque de ejercicios que el alumno deberá responder.
- Recibir la respuesta del alumno e interpretarla.
- Llevar el registro de errores y aciertos de cada alumno para posteriormente determinar la evaluación final alcanzada por ese alumno durante el estudio de algún tema o subtema.
- Establecer al ambiente de ejecución apto para aprovechar las capacidades de la microcomputadora.

5.4 Estructura del contenido temático en el sistema tutorial.

La interfase entre los subsistemas que conforman el sistema tutorial LOGMAT, permiten la identificación clara de las funciones básicas de cualquier sistema de instrucción programada. La ventaja de tener este tipo de estructura es permitir el aprovechamiento de las facilidades que cada subsistema provee.

El sistema tutorial LOGMAT es un ejemplo concreto de un sistema de instrucción programada el cual explica nociones introductorias a la lógica matemática valiéndose del empleo de un lenguaje inteligible para el alumno. En la figura 5.4.1 se muestra la tabla de lecciones, temas y subtemas que se abordan en el sistema tutorial LOGMAT.

Figura 5.4.1 Tabla de lecciones, temas y subtemas abordables en el sistema tutorial

1.	Introducción a la lógica matemática.	
	1.1	Definición de lógica matemática
	1.2	Métodos lógicos
	1.3	Lenguaje simbólico.
2	Proposiciones.	
	2.1	Definición de proposiciones
	2.2	Simples y compuestas
	2.3	Conectivos lógicos y tablas de verdad
		2.3.1 Definición de conectivos lógicos
		2.3.2 Definición de tablas de verdad
		2.3.3 Negación y doble negación
		2.3.4 Conjunción
		2.3.5 Disyunción
		2.3.6 Implicación condicional
		2.3.7 Doble condicional
	2.4	Tipos de proposiciones
		2.4.1 Tautológicas
		2.4.2 Contradictorias
		2.4.3 Contingentes
	2.5	Verdad formal y verdad empírica.
3.	Argumentos	
	3.1	Composición de un argumento
	3.2	Validez lógica de un argumento
	3.3	Leyes de implicación
		3.3.1 Modus ponendo ponens
		3.3.2 Modus tollendo tollens
		3.3.3 Modus tollendo ponens
		3.3.4 Silogismo hipotético
		3.3.5 Ley de la simplificación
		3.3.6 Ley de la conjunción
		3.3.7 Ley de la adición
	3.4	Leyes de equivalencia
	3.5	Demostración formal de la validez de un argumento

El contenido temático que se presenta en este tutorial ha sido organizado en forma modular de tal manera que los temas y subtemas son identificables como unidades instruccionales independientes facilitando de esta manera su manejo y operación.

La tabla 5.4.1 anterior muestra la estructuración propuesta del contenido temático del sistema tutorial. LOGMAT se encuentra conformado de tres lecciones principales. Definiéndose como lección la división más general que puede establecerse dentro del contenido del sistema tutorial. A su vez, cada lección se subdivide en partes que tienen un mayor grado específico: los temas. De manera que cada lección engloba a un conjunto finito de temas. Finalmente, algunos temas están compuestos de unidades instruccionales mínimas que son llamadas subtemas.

5.5 Alcance del Sistema Tutorial.

El sistema tutorial LOGMAT que se presenta en este trabajo fue diseñado para cumplir con los objetivos planteados anteriormente en la introducción, es decir, es un sistema que sirve como apoyo para la enseñanza lógica matemática y el ambiente que lo conforma es sencillo y fácil de aprender, ya que el alumno no necesita saber un lenguaje complicado ni un gran conjunto de órdenes.

LOGMAT esta enfocado a dos tipos de usuarios el profesor y el alumno. El papel del profesor es el de accesar la información de cada alumno y supervisar su aprendizaje, el papel del alumno es el ejecutar el sistema tutorial de manera que pueda aprender y ser evaluado.

Intervención del profesor dentro del sistema tutorial.

Dentro del sistema tutorial el papel del profesor es interactuar con el Subsistema de Control de Alumnos eligiendo del menú principal aquella opción que el desea ejecutar. El profesor es el responsable directo de accesar a la base de datos la información que le permitirá al alumno poder ejecutar el Subsistema de Instrucción; así mismo en relación a la información gestionada el profesor podrá consultar los avances obtenidos por cada alumno y también le dará la facilidad de omitir información de alguno de ellos cuando el criterio del profesor lo crea conveniente (Anexo B, Manual de Usuario).

Intervención del alumno dentro del sistema tutorial.

En el sistema tutorial es el alumno quien interviene directamente con el Subsistema de Instrucción el cual como se menciona anteriormente esta compuesto de temas, subtemas y ejercicios lo que en su conjunto da como resultado un sistema bien definido de instrucción asistida por computadora, cuya disposición propicia un ambiente en el cual se puede llevar a cabo un curso de lógica matemática de una manera simple y sistemática (Anexo B, Manual de Usuario).

Para utilizar cualquiera de los componentes del Subsistema de Instrucción, sólo se requiere elegir alguna de las opciones que se presentan en el menú principal e inmediatamente el alumno accesará el tema deseado. En algunos casos existen algunos temas que hacen referencia a un menú subordinado de posibles subtemas que le permiten al alumno profundizar su aprendizaje y así poder incrementar sus conocimientos. De esta manera cada una de las opciones que se enlistan en los menús representa la elección de un tema o subtema elegido por el alumno, teniendo además una serie de ejercicios secuenciales elaborados con el propósito de que el alumno reafirme lo aprendido.

5.6 Difusión del sistema tutorial.

Se sugiere difundir el sistema tutorial LOGMAT dentro de la FCA con el objeto de que esta forma de trabajo que mejore y apoye el proceso de enseñanza-aprendizaje, facilitando el estudio del alumno.

5.7 Evaluación del sistema tutorial.

Alfred Bork, observó que es poco frecuente la evaluación del software educativo, aunque sea de extrema importancia para garantizar la cualidad del producto.

Debido a la baja frecuencia de evaluación de software y a la incongruencia de las fichas de evaluación extranjeras que no se muestran adecuadas para ser aplicadas en la realidad educativa mexicana. Hemos construido y validado una ficha de evaluación de productos educativos para computadora que toma en cuenta, tanto aspectos pedagógicos, como técnicos. El objeto de la ficha es el software educativo por lo que no se justifica su aplicación a los llamados software aplicativos. A continuación, se describe el proceso de elaboración y validación de la ficha que se propone:

Elaboración de la ficha.

Para elaborar una ficha de evaluación de productos educativos para microcomputadoras fue consultada la literatura que a nuestro juicio consideramos la más pertinente, buscando verificar entre otros aspectos, los procedimientos más frecuentes utilizados en la construcción de este tipo de instrumentos.

Fueron seleccionadas dos categorías para diseñar la elaboración de la ficha: (a) de aspectos pedagógicos instruccionales y (b) de aspectos técnicos operacionales.

Cada categoría fue dividida en subcategorías, a saber: identificación del programa, contenido, metodología e interactividad (primera categoría); y documentación, interfase y recursos computacionales (segunda categoría).

Posteriormente fueron seleccionados factores que explicarán cada subcategoría en la literatura consultada de acuerdo con los siguientes criterios: (a) frecuencia de utilización de indicadores en las fichas desarrolladas por autores consultados; (b) atención a los aspectos pedagógicos.

Entre las fichas consultadas se encuentran: Campos (1985) y Lauterbach y Frey (1987).

Una vez seleccionados los factores estos fueron orientados hacia la elaboración de la versión final de la ficha de evaluación aplicable al sistema tutorial (Anexo C, Ficha de Evaluación).

Con el propósito de evaluar el sistema tutorial nos hemos basado en la aplicación de los pasos de evaluación que sugiere Sánchez. [1]

Identificamos al sistema tutorial para ser evaluado como una sugerencia didáctica que apoye al proceso de enseñanza-aprendizaje de lógica matemática, empleando la aplicación de fichas de evaluación que busquen verificar el alcance del sistema tutorial. Las respuestas que se obtendrán por cada uno de los profesores son un dato fundamental en el momento de evaluar al sistema.

---

1 Sánchez, G. Evaluación de Programas Sociales: Un enfoque a Programas Académicos Universitarios, México, UNAM, Facultad de Ingeniería, División de Estudios de Posgrado, 1994, pp. 135-138.

La evaluación del sistema tutorial tiene como objeto tomar en cuenta tanto aspectos pedagógicos, técnicos y el grado de apoyo que presta a la enseñanza ya que el objetivo que el sistema tutorial persigue es el de ayudar al alumno a aprender Lógica Matemática.

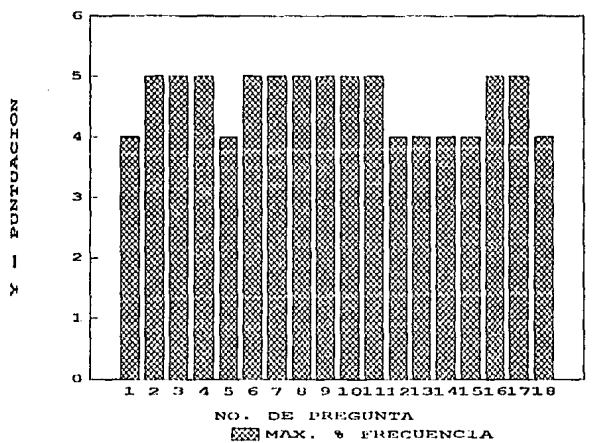
El sistema tutorial es sometido a evaluación, en condiciones reales de funcionamiento, es decir frente a profesores que dan a conocer su opinión pedagógica sobre el interés educativo del sistema tutorial (nivel escolar, grado, material necesario). Los miembros del personal académico (profesores) que conforman el grupo de evaluación juzgarán el funcionamiento del sistema y sugerirán la conveniencia de introducir algunas modificaciones en el diseño, los gráficos y la programación.

Posteriormente son comparados los propósitos iniciales mediante los cuales se elaboró el sistema tutorial contra los resultados analíticos de las fichas de evaluación y se procede a realizar de ser necesario las modificaciones señaladas por el grupo evaluador a a fin de realizar en el tutorial los ajustes pertinentes.

La intención de la evaluación no se cifra sólo en obtener una indicación de contenido, sino también de aquellos factores que pueden contribuir a mantener tanto el interés del profesor como del alumno.

Una vez realizada la evaluación ex-ante y recabada la documentación complementaria se cuenta con una visión objetiva y una valoración cualitativa ella. Los resultados generales de la evaluación señalan que más en el gráfico 5.7.1 que del 90% de las respuestas que dieron los profesores revelan que el sistema tutorial es apto, favorable y de calidad satisfactoria para ser empleado como apoyo educativo.

Figura 5.7.1 Evaluación general del sistema tutorial



6. CONCLUSIONES Y PROPUESTAS.

6.1 Conclusiones.

6.1.1 Conclusiones generales.

El uso de la computadora en la educación coadyuva al proceso de enseñanza-aprendizaje porque promueve el autoaprendizaje, por la posibilidad de usar el material durante el tiempo que cada alumno requiera, porque se puede repetir con precisión y sin agotar el sistema casi ilimitadamente y además porque este mismo sistema puede estar programado para evaluar el grado de dominio de los temas que el estudiante consulte.

La computadora tiene una particularidad que la hace un medio muy poderoso en el proceso educativo: su posibilidad de interacción con el alumno. Es este uno de los aspectos que la ponen en ventaja con respecto a otros medios como el libro, los audiovisuales, la televisión, etcétera. Además posee características que le dan grandes potencialidades: posibilidad de efectuar cálculos en forma veloz, almacenar gran cantidad de información, de elaborar y almacenar gráficos, realizar "animación", presentar diferentes secuencias de material de acuerdo con la elección del alumno así como de comunicarse con sitios remotos en forma instantánea.

Cualquier proyecto de IAC presupone una filosofía que no debe olvidarse en ningún momento del desarrollo o evaluación de un sistema educativo.

La computadora optimiza el desarrollo de las facultades cognoscitivas refiriéndose a capacidades como la resolución de problemas, dotes de reflexión y pensamiento así como técnicas relacionadas con la información como son la obtención, el análisis y la síntesis de datos.

La investigación y desarrollo de software educativo no sólo hace innovadora una nueva etapa del proceso de enseñanza-aprendizaje en la Facultad de Contaduría y Administración sino que promueve la "cultura informática" lo cual prepara también a los alumnos para desenvolverse en un medio tecnológico.

Las posibilidades de aplicación de la tecnología computacional en el sector educativo son inmensas., Sin embargo, para que ésta pueda aportar soluciones eficaces y razonables en nuestro medio, es necesario que tengamos un conocimiento de sus posibilidades de las necesidades educativas y de su contexto natural.

La elaboración de material didáctico asistido por computadora tiene severas exigencias en lo referente a la participación de expertos en la asignatura como de los recursos técnicos que están involucrados en la producción de software educativo.

El Licenciado en Informática debe promover como actividad central en el proceso instruccional el desarrollo de software educativo procurando la integración de un equipo interdisciplinario con profesionales en las otras áreas entre ellas la pedagogía y psicología educativas.

La computadora no va a suplantar al profesor sino que va a auxiliarle en su tarea informativa, adiestramiento y control. Ampliar las posibilidades de los profesores y el aprendizaje será más efectivo en determinadas asignaturas. Liberará al profesor de los aspectos mecánicos de la educación para que su tiempo, experiencia y entusiasmo se consagren a resolver los problemas educativos que son esencialmente humanos e individuales.

Deberá ser necesario preparar a los profesionales de la educación para las nuevas calificaciones y cometidos que le llevarán a seguir más de cerca la evolución de sus alumnos tanto a nivel intelectual como afectivo y comportamental al tener que abordar el desarrollo global de los mismos. El profesor debe aportar al alumno lo que la tecnología no puede: integridad, eficacia y coherencia.

6.1.2 Conclusiones específicas con respecto al sistema tutorial LOGMAT.

Como se ha mencionando a lo largo de este trabajo el sistema tutorial LOGMAT ha sido diseñado como sugerencia didáctica para apoyar el aprendizaje de la asignatura de Matemáticas I. Las características que posee LOGMAT son las de transmitir información en forma clara y concisa, formular preguntas que inciten a la investigación, propiciar que el alumno use los conocimientos recién adquiridos, a fin de reforzar la comprensión del tema o subtema visto, interpretar las respuestas del alumno, identificar los posibles errores en ellas y dar las explicaciones del caso así como favorecer el aprendizaje del alumno colocando en posición de sujeto activo.

El sistema tutorial LOGMAT es considerado un apoyo educativo apto para los profesores y un medio didáctico atractivo para los alumnos cumpliéndose de esta manera el alcance último para lo cual ha sido concebido. La utilidad de LOGMAT no se centra solamente en transmitir datos y conocimientos, sino también, resulta adecuado para desarrollar nuevas habilidades y promover procesos de pensamiento lógico invitando al alumno a conducir su propia investigación, aprendiendo así de la práctica, la acción y la experiencia de elementos que el empleo del sistema tutorial proporciona.

6.2 Futuros desarrollos.

Es importante que en México se desarrollen líneas de investigación sobre el tema de la Instrucción Asistida por Computadora, con la finalidad de que podamos ir "apropiándonos" de las nuevas tecnologías y obtener propias acerca de ello, evitando así la dependencia con otros países. En una época de grandes cambios como la actual es substancial que se tenga la iniciativa de desarrollar ideas que apoyen a la educación en nuestro país. Necesitamos tener una visión coherente acerca de lo que nos gustaría que fuera nuestra educación futura.

GLOSARIO.

Acceso.

Lectura o grabación de datos con la connotación que el contenido de dicha lectura o grabación ha tenido en cuenta. Utilizada en forma verbal, es sinónimo de entrar.

Base de datos.

Es una colección de elementos almacenados de tal manera que minimizan la redundancia.

Codificación.

Transformación de un diseño detallado en un programa real, normalmente realizado de forma automática.

Cognoscitivismo.

Se le considera como una de las teorías más avanzadas y se basa en la credibilidad esencial de entender el "esquema" o "estructura" que utiliza el cerebro para organizar el conocimiento interno.

Conductismo.

Movimiento psicológico fundado por John Watson. Esta teoría sostiene que el hombre solamente puede reaccionar a base de estímulos, de introducir información en el cerebro, de respuestas, y de los resultados de la conducta a esos estímulos.

Courseware.

Conjunto organizado de programas didácticos para el aprendizaje de alguna asignatura.

Curriculum.

Conjunto de actividades que se plancan y ejecutan bajo el control del sistema educacional, dentro o fuera del aula, pero que están diseñadas para contribuir al logro de los objetivos educativos propuestos.

Diagrama de flujo.

Representación gráfica de bajo nivel de la estructura de un programa, dando mayor importancia al flujo de control y a las acciones primarias realizadas por el programa más que a las estructuras de los datos empleados por éste.

Implementación.

Actividad de continuación desde el diseño dado de un sistema hasta una versión operativa (conocida también como una implementación) de dicho sistema, o la forma específica en que a una parte de un sistema se le hace cumplir su función. Dentro del contacto de soporte lógico informatico, el empleo del término implica, normalmente que todas las decisiones de diseño importantes se han tomado de tal manera que la actividad de implementación puede ser relativamente directa. Para muchos sistemas, varias características importantes no pueden llegar a delimitarse hasta la implementación; como ejemplo puede mencionarse el lenguaje de programación en el que se escribe el sistema, el tipo de computadora empleada, la configuración real del soporte físico o el sistema operativo utilizado.

Ingeniería de software.

Es la disciplina tecnológica y administrativa dedicada a la producción sistemática de productos de software, que son desarrollados y modificados a tiempo y dentro de un presupuesto definido. Las principales metas de esta disciplina son mejorar la calidad de estos productos y aumentar la productividad y satisfacción profesional de las personas involucradas en esta disciplina.

Instrucción Asistida por Computadora.

Término original para la utilización en línea de una computadora que administre la instrucción directamente a una o más personas. En muchas áreas, la IAC ha venido a significar ejercicios y prácticas basadas en computadora, en otras áreas, es una enseñanza basada en la computadora, particularmente la dirigida al aprendizaje del uso de una computadora o programas particulares.

Inteligencia artificial.

Capacidad de una máquina (un robot o una computadora) para imitar las acciones o habilidades humanas como la solución de problemas, la toma de decisiones, la percepción y el aprendizaje. Estudio de las computadoras y de técnicas relacionadas con ellas, tendiente a incrementar las capacidades intelectuales de las máquinas mediante una mejor programación, reconocimiento óptico de caracteres, aprendizaje artificial, síntesis de la voz, autorreparación, detección, localización y corrección automática de fallas; y autoorganización.

Interfase.

Especificación de la comunicación entre dos sistemas, dispositivos o programas.

Enseñanza Programada.

Es la presentación de la materia-objeto en forma gradual en pequeñas dosis, organizada de manera que el alumno pueda comprobar inmediatamente hasta que punto esta aprendiendo. Para lograr este objetivo, el alumno ha de participar activamente en la enseñanza escribiendo, respondiendo, hablando y practicando. También suele llamársele instrucción programada.

Lista doblemente enlazada.

Lista encadenada donde cada elemento se encuentra enlazado tanto a un predecesor como a un sucesor. De esta manera se puede atravesar la lista en doble dirección. La flexibilidad que da el doble enlace se logra por el contenido de la cabecera de almacenamiento, el posicionamiento y puesta a cero de los enlaces extras.

Lenguaje de autor.

Software que posibilita la creación de material didáctico con un número limitado de instrucciones de programación, reduciendo al mínimo la preocupación del profesor por la técnica informática. Un lenguaje de autor, en cuanto al nivel de complejidad de sus posibilidades no difiere mucho de un lenguaje de programación.

Lógica matemática.

Ciencia del razonamiento, la prueba, el pensamiento o la deducción. En la lógica matemática la investigación comprende métodos matemáticos tomados del álgebra o de la teoría de algoritmos. Los dos sistema comunes son el cálculo proposicional y el de predicados.

Menú.

Es una lista que aparece en pantalla dentro de la que se puede seleccionar un elemento. La lista de opciones puede visualizarse con un solo código para cada una de ellas. La selección puede usarse pulsando la tecla correspondiente.

Neoconductismo.

Una versión moderna del conductismo que se preocupa cada vez más por los procesos complejos humanos. Los neoconductistas insisten en plantear preguntas precisas y bien delineadas, recurriendo al del método científico y llevando a cabo una investigación cuidadosa y exacta.

Sistema de autor.

Es un conjunto de programas compatibles entre sí, que permiten crear programas didácticos sin saber informática y se caracteriza por una mayor sencillez en las instrucciones, que se presentan en forma de preguntas u opciones en lugar de escribirse en forma de código y por un nivel de interacción más próximo al usuario que al programador.

Pedagogía.

Disciplina que estudia los principios y métodos de enseñanza y educación. La enseñanza es el proceso mediante el cual se imparten conocimientos a un individuo. La educación en sentido general, es la asimilación de la cultura del grupo en que se vive, así como el estudio de los valores culturales de la humanidad. Es una función social, y tiene los siguientes objetivos: familiarizar a una nueva generación con los valores culturales de la anterior; inculcar al alumno los ideales, hábitos y creencias de su sociedad; formar en él una personalidad sana que redunde en su propio beneficio y forme un individuo útil al grupo social y al resto del mundo.

Preprosesador.

Dispositivo que coloca los registros fuente (datos) en un formato que facilita el procesamiento de algún sistema en una computadora.

Programa de autor.

Herramienta informática diseñada en todas sus partes por profesionales con experiencia en programación el cual sólo debe determinar ciertos parámetros de uso, respondiendo a una serie de preguntas que hace el programa, e introducir los datos (texto, preguntas, etc.) relativos al programa de aplicación que se desea crear. Es la forma más sencilla de iniciarse en el desarrollo del software didáctico.

Programación modular.

Estilo de programación en que el programa completo se descompone en un conjunto de componentes, denominados módulos, cada uno de los cuales tiene un tamaño manejable, una finalidad y una relación bien definida, con el mundo exterior.

Prototipo.

Desarrollo de la versión preliminar de un sistema lógico informatico, con la finalidad de permitir la investigación de ciertos aspectos del sistema. Con frecuencia, el propósito principal de un prototipo es la obtención de la retroalimentación desde los usuarios potenciales; la especificación de los requisitos del sistema puede actualizarse después para reflejar dicha retroalimentación y de esta manera aumentar la confianza en el sistema final. De forma adicional (o alternativa), puede utilizarse un prototipo para investigar áreas de problemas particulares, o algunas implicaciones de diseño alternativo o de decisiones de implementación. Por regla general, la intención de realizar un prototipo es la de obtener la información necesaria lo más rápidamente posible; con la menor inversión de recursos, siendo, por lo tanto, normal concentrarse en ciertos aspectos del sistema que se intenten desarrollar e ignorar otros por completo. Por ejemplo, puede desarrollarse un prototipo sin atender a su eficacia o rendimiento y pueden omitirse enteramente algunas funciones del sistema. Sin embargo se debe ser realista en los aspectos específicos que se están investigando.

Anexo A

Ejemplo de código fuente.

```
/*****************************************************************************/
/* Programa:  LOGMAT.C                                                       */
/*                                                                           */
/* Sistema :  SISTEMA TUTORIAL DE LOGICA MATEMATICA                          */
/*            (LOGMAT)                                                       */
/* Autor   :  Maribel Asencio Armenta                                        */
/* Ultima modificación: Julio 1994                                           */
/* Descripción: Programa principal del Subsistema de Instrucción             */
/* Programa llamado por..                                                    */
/* Programa que llama a.. A los temas y subtemas                             */
/* Parametros que recibe:                                                    */
/*****************************************************************************/

# include "stdio.h"
# include "stdlib.h"
# include "alloc.h"
# include "string.h"
# include "graphics.h"
# include "process.h"
# include "ctype.h"
# include "conio.h"
# include "pintura.h"
# define CORRECTO 100
# define INCORRECTO -1

struct alumno{
char nolista        [3];
char nombre         [45];
char nocuenta       [9];
char cldeflog       [3];
char clmetlog       [3];
char cllensim       [3];
char cldefpro       [3];
char clsimcom       [3];
char cldefcon       [3];
char cldeftav       [3];
char clnegdne       [3];
char clconjun       [3];
char cldisyun       [3];
char climpcon       [3];
char cldobcon       [3];
char cltautol       [3];
char clcontra       [3];
char clcontin       [3];
char clvfrvem       [3];
char clcomarg       [3];
char clvaloar       [3];
char clmopopo       [3];
char clmototo       [3];
char clmotopo       [3];
char clsilhip       [3];
char clleysim       [3];
char clleycon       [3];
char clleyndi       [3];
char clleyequ       [3];
char cldefvar       [3];
struct alumno *siguiente;
struct alumno *anterior;
} entrada_lista;
```

```
struct alumno *principio;
struct alumno *final;
struct alumno *hallar(char *);

void introducir(void), guardar(void);
void cargar(void), listado(void);
void borrar(struct alumno **, struct alumno **);
void almacenar_dco(struct alumno *i, struct alumno **principio, struct alumno **final);
void entrada(char *, char *, int), mostrar(struct alumno *), inicio(struct alumno *);

/*VARIABLES GLOBALES*/
char np[3]="NP", px[3]="PX", cad[2], *mulo, caletra[3];

union tecla{
        int i;
        char c[2];
        }t, o;
int opcion=0, respsn=0, calnum=0;

main( )
{
principio=final=NULL;
clrscr( );
spawnl(P_WAIT,"MARK.COM",NULL);
spawnl(P_WAIT,"COMANECI.EXE",NULL);
spawnl(P_WAIT,"ST.EXE"," "," ","UNAM-FCA.SH~",NULL);
spawnl(P_WAIT,"ST.EXE"," "," ","INICIO.SH~",NULL);
spawnl(P_WAIT,"RELEASE.EXE",NULL);
estudiante( );
nosound( );
clrscr( );
}

/*PRESENTAR EL MENU DE LECCIONES */
void inicio(struct alumno *info)
{
int bandera=0, primero=0;
 do
  {
  bandera=primero=0;
  abre( );
  pintura("MENU1.PIC");
  opcion=mensaje(4,0,165);
  cierra( );
  switch(opcion)
  {
        case 1: bandera=segundo(primero,info);
                break;
        case 2: bandera=tercero(primero,info);
                break;
        case 3: bandera=cuarto(primero,info);
                break;
        case 4: cierra( );
                exit(1);
                break;
  }
 }
 while (bandera==0);
}
```

```
/* PRESENTAR EL MENU CON LOS TEMAS DE LA LECCION 1. INTRODUCCION A LOGICA */
/* MATEMATICA */
segundo(int first, struct alumno *info)
{
 int bandera2=0;
  do
   {
   bandera2=0;
   abre( );
   pintura("MENU2.PIC");
   opcion=mensaje(4,0,165);
   cierra( );
   switch(opcion){
         case 1: /* 1.1 DEFINICION DE LOGICA MATEMATICA */
                  if (strcmp(np,info->ldeflog))
                  respsn=ventana(respsn);
                   if((respsn==1) || (!strcmp(np,info->ldeflog)))
                    {
                    calnum=spawnl(P_WAIT,"SIDEFLOG.EXE","SIDEFLOG.EXE","1",calnum,NULL);
                           if(calnum==INCORRECTO)
                           {
                           notema("SIDEFLOG.EXE");
                           bandera2=0;
                           break;
                           }
                           if(calnum!=CORRECTO)
                           {
                           *caletra=calcular(caletra,calnum);
                           strcpy(info->ldeflog,caletra);
                           guardar( );
                           }
                    }
                  bandera2=0;
                  break;
         case 2: /* 1.2 METODOS LOGICOS */
                  if (strcmp(np,info->lmetlog))
                  respsn=ventana(respsn);
                   if((respsn==1) || (!strcmp(np,info->lmetlog)))
                    {
                    calnum=spawnl(P_WAIT,"SIMETLOG.EXE","SIMETLOG.EXE","1",calnum,NULL);
                           if(calnum==INCORRECTO)
                           {
                           notema("SIMETLOG.EXE");
                           bandera2=0;
                           break;
                           }
                           if(calnum!=CORRECTO)
                           {
                           *caletra=calcular(caletra,calnum);
                           strcpy(info->lmetlog,caletra);
                           guardar( );
                           }
                    }
                  bandera2=0;
                  break;
         case 3: /* 1.3 LENGUAJE SIMBOLICO */
                  if (strcmp(np,info->llensim))
                  respsn=ventana(respsn);
                   if((respsn==1) || (!strcmp(np,info->llensim)))
```

```
                {
                calnum=spawnl(P_WAIT,"SILENSIM.EXE","SILENSIM.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SILENSIM.EXE");
                        bandera2=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->clle.sim,caletra);
                        guardar( );
                        }
                }
                bandera2=0;
                break;
        case 4: first=0;
                return first;
 }
 }
 while (bandera2==0);
}

/* PRESENTAR EL MENU CON LOS TEMAS DE LA LECCION 2. PROPOSICIONES */
terceo(int secc_a,struct alumno *info)
{
 int bande:a3=0, terc=0;
  do
  {
   oendera3=terc=0;
   abre( );
   pintura("MENU3.PIC");
   opcion=mensaje(6,0,175);
   cierra( );
   switch(opcion){
        case 1: /* 2.1 DEFINICION DE PROPOSICIONES */
                if (strcmp(np,info->cldefpro))
                respsn=ventana(respsn);
                 if((respsn==1) || (!strcmp(np,info->cldefpro)))
                  {
                  calnum=spawnl(P_WAIT,"SIDEFPRO.EXE","SIDEFPRO.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SIDEFPRO.EXE");
                        bandera3=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->cldefpro,caletra);
                        guardar( );
                        }
                  }
                bandera3=0;
                break;
        case 2: /* 2.2 SIMPLES-COMPUESTAS */
                if (strcmp(np,info->clsimcom))
                respsn=ventana(respsn);
```

```
                if((respsn==1) || (!strcmp(np,info->clsimcom)))
                 {
                 calnum=spawnl(P_WAIT,"SISIMCOM.EXE","SISIMCOM.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SISIMCOM.EXE");
                        bandera3=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->clsimcom,caletra);
                        guardar( );
                        }
                 }
                bandera3=0;
                break;
        case 3: /* 2.3 CONECTIVOS LOGICOS Y TABLAS DE VERDAD */
                bandera3=tercera(terc,info);
                break;
        case 4: /* 2.4 TIPOS DE PROPOSICIONES */
                bandera3=tercerb(terc,info);
                break;
        case 5: /* 2.5 VERDAD FORMAL Y EMPIRICA */
                if (strcmp(np,info->clvfrvem))
                respsn=ventana(respsn);
                 if((respsn==1) || (!strcmp(np,info->clvfrvem)))
                  {
                  calnum=spawnl(P_WAIT,"SIVFRVEM.EXE","SIVFRVEM.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SIVFRVEM.EXE");
                        bandera3=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->clvfrvem,caletra);
                        guardar( );
                        }
                 }
                bandera3=0;
                break;
        case 6: second=0;
                return second;
  }
 }
 while (bandera3==0);
}

/* PRESENTAR EL MENU SUBORDINADO CON LOS SUBTEMAS DE LA LECCION*/
/* 2. PROPOSICIONES, 2.3 CONECTIVOS LOGICOS Y TABLAS DE VERDAD */
tercera(int threea, struct alumno *info)
{
 int band3a=0;
 do
  {
  band3a=0;
```

```
abre( );
pintura("MENU7.PIC");
opcion=mensaje(8,0,180);
cierra( );
switch(opcion){
        case 1: /* 2.3 1 DEFINICION DE CONECTIVOS LOGICOS */
                if (strcmp(np,info->cldefcon))
                respsn=ventana(respsn);
                 if((respsn==1) || (!strcmp(np,info->cldefcon)))
                  {
                  calnum=spawnl(P_WAIT,"SIDEFCON.EXE","SIDEFCON.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SIDEFCON.EXE");
                        band3a=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->cldefcon,caletra);
                        guardar( );
                        }
                  }
                band3a=0;
                break;
        case 2: /* 2.3.2 DEFINICION DE TABLAS DE VERDAD */
                if (strcmp(np,info->cldeftav))
                respsn=ventana(respsn);
                 if((respsn==1) || (!strcmp(np,info->cldeftav)))
                  {
                  calnum=spawnl(P_WAIT,"SIDEFTAV.EXE","SIDEFTAV.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SIDEFTAV.EXE");
                        band3a=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->cldeftav,caletra);
                        guardar( );
                        }
                  }
                band3a=0;
                break;
        case 3: /* 2.3.3 NEGACION Y DOBLE NEGACION */
                if (strcmp(np,info->clnegdne))
                respsn=ventana(respsn);
                 if((respsn==1) || (!strcmp(np,info->clnegdne)))
                  {
                  calnum=spawnl(P_WAIT,"SINEGDNE.EXE","SINEGDNE.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SINEGDNE.EXE");
                        band3a=0;
                        break;
                        }
```

```
                if(calnum!=CORRECTO)
                {
                *caletra=calcular(caletra,calnum);
                strcpy(info->clnegdne,caletra);
                guardar( );
                }
          }
        band3a=0;
        break;
case 4: /* 2.3.4 CONJUNCION */
        if (strcmp(np,info->clconjun))
        respsn=ventana(respsn);
         if((respsn==1) || (!strcmp(np,info->clconjun)))
          {
          calnum=spawnl(P_WAIT,"SICONJUN.EXE","SICONJUN.EXE","1",calnum,NULL);
                if(calnum==INCORRECTO)
                {
                notema("SICONJUN.EXE");
                band3a=0;
                break;
                }
                if(calnum!=CORRECTO)
                {
                *caletra=calcular(caletra,calnum);
                strcpy(info->clconjun,caletra);
                guardar( );
                }
          }
        band3a=0;
        break;
case 5: /* 2.3.5 DISYUNCION */
        if (strcmp(np,info->cldisyun))
        respsn=ventana(respsn);
         if((respsn==1) || (!strcmp(np,info->cldisyun)))
          {
          calnum=spawnl(P_WAIT,"SIDISYUN.EXE","SIDISYUN.EXE","1",calnum,NULL);
                if(calnum==INCORRECTO)
                {
                notema("SIDISYUN.EXE");
                band3a=0;
                break;
                }
                if(calnum!=CORRECTO)
                {
                *caletra=calcular(caletra,calnum);
                strcpy(info->cldisyun,caletra);
                guardar( );
                }
          }
        band3a=0;
        break;
case 6: /* 2.3.6. IMPLICACION CONDICIONAL */
        if (strcmp(np,info->climpcon))
        respsn=ventana(respsn);
         if((respsn==1) || (!strcmp(np,info->climpcon)))
          {
          calnum=spawnl(P_WAIT,"SIIMPCON.EXE","SIIMPCON.EXE","1",calnum,NULL);
                if(calnum==INCORRECTO)
                {
```

```
                        notema("SIIMPCON.EXE");
                        band3a=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->climpcon,caletra);
                        guardar( );
                        }
                }
                band3a=0;
                break;
        case 7:/* 2.3.7 DOBLE CONDICIONAL */
                if (strcmp(np,info->cldobcon))
                respsn=ventana(respsn),
                 if((respsn==1) || (!strcmp(np,info->cldobcon)))
                  {
                  calnum=spawnl(P_WAIT,"SIDOBCON.EXE","SIDOBCON.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SIDOBCON.EXE");
                        band3a=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->cldobcon,caletra);
                        guardar( );
                        }
                }
                band3a=0;
                break;
        case 8: threea=0;
                return threea;
  }
  }
 while (band3a==0);
}

/* PRESENTAR EL MENU SUBORDINADO CON LOS SUBTEMAS DE LA LECCION */
/* 2. PROPOSICIONES, 2.4 TIPOS DE PROPOSICIONES */
tercerb(int threeb, struct alumno *info)
{
 int band3b=0;
  do
   {
   band3b=0;
   abre( );
   pintura("MENU8.PIC");
   opcion=mensaje(4,0,167);
   cierra( );
   switch(opcion){
        case 1:/* 2.4.1 TAUTOLOGICAS */
                if (strcmp(np,info->cltautol))
                respsn=ventana(respsn);
                 if((respsn==1) || (!strcmp(np,info->cltautol)))
                  {
```

```
            calnum=spawnl(P_WAIT,"SITAUTOL.EXE","SITAUTOL.EXE","1",calnum,NULL);
                    if(calnum==INCORRECTO)
                    {
                    notema("SITAUTOL.EXE");
                    band3b=0;
                    break;
                    }
                    if(calnum!=CORRECTO)
                    {
                    *caletra=calcular(caletra,calnum);
                    strcpy(info->clsautol,caletra);
                    guardar( );
                    }
              }
            band3b=0;
            break;
case 2: /* 2.4.2 CONTRADICTORIAS */
            if (strcmp(np,info->clcontra))
            respsn=ventana(respsn);
             if((respsn==1) || (!strcmp(np,info->clcontra)))
              {
              calnum=spawnl(P_WAIT,"SICONTRA.EXE","SICONTRA.EXE","1",calnum,NULL);
                    if(calnum==INCORRECTO)
                    {
                    notema("SICONTRA.EXE");
                    band3b=0;
                    break;
                    }
                    if(calnum!=CORRECTO)
                    {
                    *caletra=calcular(caletra,calnum);
                    strcpy(info->clcontra,caletra);
                    guardar( );
                    }
              }
            band3b=0;
            break;
case 3: /* 2.4.3 CONTIGENTES */
            if (strcmp(np,info->clcontin))
            respsn=ventana(respsn);
             if((respsn==1) || (!strcmp(np,info->clcontin)))
              {
              calnum=spawnl(P_WAIT,"SICONTIN.EXE","SICONTIN.EXE","1",calnum,NULL);
                    if(calnum==INCORRECTO)
                    {
                    notema("SICONTIN.EXE");
                    band3b=0;
                    break;
                    }
                    if(calnum!=CORRECTO)
                    {
                    *caletra=calcular(caletra,calnum);
                    strcpy(info->clcontin,caletra);
                    guardar( );
                    }
              }
            band3b=0;
            break;
case 4: threeb=0;
```

```
                return threeb;
 }
 }
while (band3b==0);
}

/* PRESENTAR EL MENU CON LOS TEMAS DE LA LECCION 3. ARGUMENTOS */
cuarto(int three, struct alumno *info)
{
int bandera4=0, cuart=0;
do
 {
 bandera4=cuart=0;
 abre( );
 pintura("MENU4.PIC");
 opcion=mensaje(6,0,180);
 cierra( );
 switch(opcion){
        case 1: /* 3.1 COMPOSICION DE UN ARGUMENTO */
                if (strcmp(np,info->clcomarg))
                respsn=ventana(respsn);
                 if((respsn==1) || (!strcmp(np,info->clcomarg)))
                  {
                  calnum=spawnl(P_WAIT,"SICOMARG.EXE","SICOMARG.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SICOMARG.EXE");
                        bandera4=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->clcomarg,caletra);
                        guardar( );
                        }
                  }
                bandera4=0;
                break;
        case 2: /* 3.2 VALIDEZ LOGICA DE UN ARGUMENTO */
                if (strcmp(np,info->clvaloar))
                respsn=ventana(respsn);
                 if((respsn==1) || (!strcmp(np,info->clvaloar)))
                  {
                  calnum=spawnl(P_WAIT,"SIVALOAR.EXE","SIVALOAR.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SIVALOAR.EXE");
                        bandera4=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->clvaloar,caletra);
                        guardar( );
                        }
                  }
                bandera4=0;
```

```
                break;
        case 3: /* 3.3 LEYES DE IMPLICACION */
                bandera4=quinto(cuart,info);
                break;
        case 4: /* 3.4 LEYES DE EQUIVALENCIA */
                bandera4=sexto(cuart,info);
                break;
        case 5: /* 3.5 DEMOSTRACION FORMAL DE LA VALIDEZ DE ARGUMENTOS */
                if (strcmp(np,info->cldefvar))
                respsn=ventana(respsn);
                 if((respsn==1) || (!strcmp(np,info->cldefvar)))
                  {
                  calnum=spawnl(P_WAIT,"SIDEFVAR.EXE","SIDEFVAR.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SIDEFVAR.EXE");
                        bandera4=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->cldefvar,caletra);
                        guardar( );
                        }
                  }
                bandera4=0;
                break;
        case 6: three=0;
                return three;
 }
 }
 while (bandera4==0);
}

/* PRESENTAR EL MENU CON LOS SUBTEMAS DE LA LECCION 3. ARGUMENTOS ,*/
/* 3.3 LEYES DE IMPLICACION */
quinto(int forth, struct alumno *info)
{
 int bandera5=0;
  do
   {
   bandera5=0;
   abre( );
   pintura("MENU5.PIC");
   opcion=mensaje(8,0,175);
   cierra( );
   switch(opcion){
        case 1: /* 3.3.1 MODUS PONENDO PONES */
                if (strcmp(np,info->clmopopo))
                respsn=ventana(respsn);
                 if((respsn==1) || (!strcmp(np,info->clmopopo)))
                  {
                  calnum=spawnl(P_WAIT,"SIMOPOPO.EXE","SIMOPOPO.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SIMOPOPO.EXE");
                        bandera5=0;
                        break;
                        }
```

```
                    if(calnum!=CORRECTO)
                    {
                    *caletra=calcular(caletra,calnum);
                    strcpy(info->clmopopo,caletra);
                    guardar( );
                    }
                }
            bandera5=0;
            break;
case 2: /* 3.3.2 MODUS TOLLENDO TONENS */
            if (strcmp(np,info->clmototo))
            respsn=ventana(respsn);
              if((respsn==1) || (!strcmp(np,info->clmototo)))
               {
               calnum=spawnl(P_WAIT,"SIMOTOTO.EXE","SIMOTOTO.EXE","1",calnum,NULL);
                    if(calnum==INCORRECTO)
                    {
                    notema("SIMOTOTO.EXE");
                    bandera5=0;
                    break;
                    }
                    if(calnum!=CORRECTO)
                    {
                    *caletra=calcular(caletra,calnum);
                    strcpy(info->clmototo,caletra);
                    guardar( );
                    }
                }
            bandera5=0;
            break;
case 3: /* 3.3.3 MODUS TOLLENDO PONENS */
            if (strcmp(np,info->clmotopo))
            respsn=ventana(respsn);
              if((respsn==1) || (!strcmp(np,info->clmotopo)))
               {
               calnum=spawnl(P_WAIT,"SIMOTOPO.EXE","SIMOTOPO.EXE","1",calnum,NULL);
                    if(calnum==INCORRECTO)
                    {
                    notema("SIMOTOPO.EXE");
                    bandera5=0;
                    break;
                    }
                    if(calnum!=CORRECTO)
                    {
                    *caletra=calcular(caletra,calnum);
                    strcpy(info->clmotopo,caletra);
                    guardar( );
                    }
                }
            bandera5=0;
            break;
case 4: /* 3.3.4 SILOGISMO HIPOTETICO */
            if (strcmp(np,info->clsilhip))
            respsn=ventana(respsn);
              if((respsn==1) || (!strcmp(np,info->clsilhip)))
               {
               calnum=spawnl(P_WAIT,"SISILHIP.EXE","SISILHIP.EXE","1",calnum,NULL);
                    if(calnum==INCORRECTO)
                    {
```

```
                    notema("SISILHIP..EXE");
                    bandera5=0;
                    break;
                    }
                    if(calnum!=CORRECTO)
                    {
                    *caletra=calcular(caletra,calnum);
                    strcpy(info->clsilhip,caletra);
                    guardar( );
                    }
            }
          bandera5=0;
          break;
case 5: /* 3.3.5 LEY DE LA SIMPLIFICACION */
          if (strcmp(np,info->clleysim))
          respsn=ventana(respsn);
           if((respsn==1) || (!strcmp(np,info->clleysim)))
            {
            calnum=spawnl(P_WAIT,"SILEYSIM.EXE","SILEYSIM.EXE","1",calnum,NULL);
                    if(calnum==INCORRECTO)
                    {
                    notema("SILEYSIM.EXE");
                    bandera5=0;
                    break;
                    }
                    if(calnum!=CORRECTO)
                    {
                    *caletra=calcular(caletra,calnum);
                    strcpy(info->clleysim,caletra);
                    guardar( );
                    }
            }
          bandera5=0;
          break;
case 6: /* 3.3.6 LEY DE LA CONJUNCION */
          if (strcmp(np,info->clleycon))
          respsn=ventana(respsn);
           if((respsn==1) || (!strcmp(np,info->clleycon)))
            {
            calnum=spawnl(P_WAIT,"SILEYCON.EXE","SILEYCON.EXE","1",calnum,NULL);
                    if(calnum==INCORRECTO)
                    {
                    notema("SILEYCON.EXE");
                    bandera5=0;
                    break;
                    }
                    if(calnum!=CORRECTO)
                    {
                    *caletra=calcular(caletra,calnum);
                    strcpy(info->clleycon,caletra);
                    guardar( );
                    }
            }
          bandera5=0;
          break;
case 7: /* 3.3.7 LEY DE LA ADICION */
          if (strcmp(np,info->clleyadi))
          respsn=ventana(respsn);
           if((respsn==1) || (!strcmp(np,info->clleyadi)))
```

```
                {
                calnum=spawnl(P_WAIT,"SILEYADI.EXE","SILEYADI.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SILEYADI.EXE");
                        bandera5=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->clleyadi,caletra);
                        guardar( );
                        }
                }
                bandera5=0;
                break;
        case 8: forth=0;
                return forth;
 }
 }
 while (bandera5==0);
}

/* PRESENTAR EL MENU CON LOS SUBTEMAS DE LA LECCION 3.ARGUMENTOS, */
/* 3.4 LEYES DE EQUIVALENCIA */
sexto(int five, struct alumno *info)
{
 int bandera6=0;
 do
 {
  bandera6=0;
  abre( );
  pintura("MENU6.PIC");
  opcion=mensaje(3,0,170);
  cierra( );
  switch(opcion){
        case 1: /* 3.4 LEYES DE EQUIVALENCIA */
                if (strcmp(np,info->clleyequ))
                respsn=ventana(respsn);
                 if((respsn==1) || (!strcmp(np,info->clleyequ)))
                 {
                 calnum=spawnl(P_WAIT,"SILEYEQU.EXE","SILEYEQU.EXE","1",calnum,NULL);
                        if(calnum==INCORRECTO)
                        {
                        notema("SILEYEQU.EXE");
                        bandera6=0;
                        break;
                        }
                        if(calnum!=CORRECTO)
                        {
                        *caletra=calcular(caletra,calnum);
                        strcpy(info->clleyequ,caletra);
                        guardar( );
                        }
                }
                bandera6=0;
                break;
        case 2: five=0,
                return five;
```

```
  }
 }
 while (bandera6==0);
}

/* VALIDAR LA OPCION */
mensaje (opcionm, resp, x)
int opcionm, resp, x;
{
char *buffito;
char *numero="0";
int pon=0;
buffito=(char *)malloc(5000);
        getimage(134,x-2,185,x+10,buffito);
        *numero=toupper(getch( ));
      setfillstyle(SOLID_FILL,BLACK);
        bar(154,x-2,162,x+9);
        setcolor(BLUE);
        outtextxy(155,x,numero);
        pon=atoi(numero);
        delay(2000);
        resp=pon;
        if (pon<=0 || pon>opcionm)
        {
        do
        {
        putimage(134,x-2,buffito,COPY_PUT);
        sound(500);
        delay(100);
        nosound( );
        setfillstyle(SOLID_FILL,BLACK);
        bar(138,x-2,180,x+9);
        outtextxy(140,x,"ERROR");
        delay(1000);
        putimage(134,x-2,buffito,COPY_PUT);
        *numero=toupper(getch( ));
        setfillstyle(SOLID_FILL,BLACK);
        bar(154,x-2,162,x+9);
        outtextxy(155,x,numero);
        pon=atoi(numero);
        delay(2000);
        resp=pon;
        }
        while (pon<=0 || pon>opcionm);
        free(buffito);
        return resp;
        }
        else
        free(buffito);
        return resp;
}

/* ACTIVAR MODO GRAFICO */
abre( )
{
 int grdriver=CGA, grmode=3;
 initgraph(&grdriver,&grmode,");
}
/* DESACTIVAR MODO GRAFICO */
cierra( )
```

```
{
closegraph( );
restorecrtmode( );
}

/* PRESENTAR GRAFICO PARA VALIDAR EL ACCESO AL ALUMNO */
estudiante( )
{
struct alumno *info, *hallar( );
char pupil[30], nombre[15], apepat[15], apemat[15];
abre( );
pintura("ALUMNO.PIC");
cargar( );
*nombre=lee(nombre,9,35);
*apepat=lee(apepat,9,75);
*apemat=lee(apemat,9,115);
strcat(apepat," ");
strcat(apemat," ");
strcat(apemat,nombre);
strcat(apepat,apemat);
strcpy(pupil,apepat);
delay(1000);
busqueda(pupil);
cierra( );
}

/* LEER EL NOMBRE COMPLETO DEL ALUMNO */
lee(char dato[10],int x,int y)
{
char answer[2]=" ";
int menos=2, count=0, indice1=0, continua=0;
   char *buffo;
   buffo=(char *)malloc(5000);
   getimage(x,y,x+20,y+14,buffo);

/* ELIMINA EL PRIMER BLANCO DE *DATO */
for(indice1=0;indice1;indice1++)
         {
         cad[indice1]=dato[indice1];
         }
strcpy(dato,cad);
         setfillstyle(SOLID_FILL,BLACK);
         bar(x,y,x+20,y+14);
do{
 t.i=bioskey (1);
 t.i=bioskey (0);
         if (t.c[0]==8) /* TECLA BACK SPACE PARA BORRAR */
         {
         char cad[10];
         putimage(x-20,y,buffo,COPY_PUT);
         x=x-40;
                  if (count!=1)
                  {
                           for(indice1=0;indice1<=count-menos;indice1++)
                           {
                           cad[indice1]=dato[indice1];
                           }
                           strcpy(dato,cad);
                           count=count-1;
                  }
```

```
            else
            {
                    sound(500);
                    delay(100);
                    nosound( );
                    count=0;
                    menos=0;
                    for(indice1=0;indice1;indice1++)
                    {
                    nulo[indice1]=dato[indice1];
                    }
                    strcpy(dato,nulo);
            }
    }
     if (t.c[0]!=27 && t.c[0]!=8 && t.c[0]!=13)
     {
     *answer=toupper(t.c[0]);
     setfillstyle(SOLID_FILL,BLACK);
     bar(x,y,x+20,y+14);
     setcolor(BLUE);
     outtextxy(x+7,y+3,answer);
     strcat(dato,answer);
   count=count+1;
     continua=1;
     }
     if(t.c[0]==27)    /* TECLA ESC PARA SALIR */
     {
     continua=vescape(continua);
     if (continua==0)
     {
     cierra( );
     spawnl(P_WAIT,"RELEASE.EXE",NULL);
     nosound( );
     clrscr( );
     exit(1);
     }
     else
     x=x-20;
     }
     if(t.c[0]==13)
     {
     sound(800);
     delay(150);
     sound(900);
     delay(100);
     sound(1000);
     delay(100);
     nosound( );
     continua=0;
     }
     x=x+20;
}
while(continua==1 && count<=14);
indice1=0;
count=0;
return *dato;
}

/* DESPLEGAR UNA VENTANA DE ESCAPE */
vescape(sigue)
```

```
int sigue;
  {
   char *buffo, *sn=" ", *si="S", *no="N";
   buffo=(char *)malloc(5000);
   getimage(45,45,285,101,buffo);
   setfillstyle(SOLID_FILL,BLACK);
   bar(46,46,284,109);
   setcolor(2);
   settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
   outtextxy(55,55,"EL TUTORIAL FUE INTERRUMPIDO");
   outtextxy(95,65,"DESEAS CONTINUAR?");
   outtextxy(130,80,"SI   NO");
   rectangle(48,48,282,98);
   rectangle(126,75,146,92);
   rectangle(175,75,195,92);
o.i=bioskey (1);
o.i=bioskey (0);
*sn=toupper(o.c[0]);
if((*sn!=*si)&&(*sn!=*no))
 {
        do
        {
        sound(1240);
        delay(100);
        nosound( );
        o.i=bioskey (1);
        o.i=bioskey (0);
        *sn=toupper(o.c[0]);
        }
    while((*sn!=*si)&&(*sn!=*no));
 }
if(*sn==*si)
   {
   putimage(45,45,buffo,COPY_PUT);
   sigue=1;
   return sigue;
   }
if (*sn==*no)
 {
 sigue=0;
 return sigue;
 }
}

/* BUSCAR EL NOMBRE DEL ALUMNO EN LA BASE DE DATOS */
struct alumno *hallar(char *nombre)
{
 struct alumno *info;
 info=principio;
 while(info) {
  if(!strcmp(nombre,info->nombre)) return info;
  info=info->siguiente;
 }
    setfillstyle(SOLID_FILL,2);
    bar(46,46,284,110);
    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
    setcolor(WHITE);
    outtextxy(96,52,"PERMISO DENEGADO");
    setcolor(BLACK);
    outtextxy(65,65,"NO SE ENCUENTRA TU NOMBRE");
```

```
   outtextxy(85,75,"EN LA BASE DE DATOS");
   outtextxy(56,85,"ES NECESARIO QUE INTERVENGA");
   outtextxy(115,95,"EL PROFESOR");
   rectangle(48,48,282,108);
 return (NULL);
}

/* CREAR LISTA DOBLEMENTE ENLAZADA EN ORDEN */
 void almacenar_doc(
 struct alumno *i,
 struct alumno **principio,
 struct alumno **final
 )
 {
 struct alumno *original, *p;
  if(*final==NULL) {
  i->siguiente=NULL;
  i->anterior=NULL;
  *final = i;
  *principio =i,
  return;
  }
 p=*principio;
 original=NULL;
 while(p)
 {
  if(strcmp(p->nombre, i->nombre)<0){
  original=p;
  p=p->siguiente;
  }
  else {
   if(p->anterior) {
   p->anterior->siguiente = i;
   i->siguiente = p;
   i->anterior= p->anterior;
   p->anterior=i;
   return;
                }
   i->siguiente = p;
   i->anterior = NULL;
   p->anterior = i;
   *principio = i;
   return;
  }
  }
  original->siguiente = i;
  i->siguiente = NULL;
  i->anterior = original;
  *final=i;
 }

 /* INICIALIZAR EL APUNTADOR EN EL PRIMER ELEMENTO DE LA BASE DE DATOS Y */
 /* BUSCAR EL NOMBRE DEL ALUMNO EN ELLA */
 busqueda(name)
 char *name;
  {
   struct alumno *info, *hallar( );
   info=hallar(name);
   if(!info)
```

```
 {
 escapa( );
 }
 else
 cierra( );
 inicio(info);
}

/* FUNCION DE ESCAPE*/
escapa( )
{
   setfillstyle(SOLID_FILL,2);
   bar(46,115,284,130);
   settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
   setcolor(BLACK);
   outtextxy(70,119,"TECLEA [ESC] PARA SALIR");
   rectangle(48,117,282,128);
do{
o.i=bioskey (1);
o.i=bioskey (0);
if (o.c[0]==27){ cierra( ); exit(1);}
else
sound(500);
delay(100);
nosound( );
} while(o.c[0]!=27);
}

/* CARGAR EN MEMORIA LA BASE DE DATOS */
void cargar( )
{
struct alumno *info;
FILE *fp;
fp=fopen("alumnos.dat","rb");
if(!fp){
   setfillstyle(SOLID_FILL,2);
   bar(46,50,284,100);
   setcolor(BLACK);
   settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
   outtextxy(60,58,"NO SE ENCUENTRA DISPONIBLE");
   outtextxy(58,70,"LA BASE DE DATOS PARA PODER");
   outtextxy(70,82,"UTILIZAR ESTE TUTORIAL");
   rectangle(48,52,282,98);
   escapa( );
}
while(principio){
 info=principio->siguiente;
 free(info);
 principio=info;
 }
 principio=final=NULL;
while(!feof(fp)){
 info=(struct alumno *) malloc(sizeof(struct alumno));
 if(!info) {
   setfillstyle(SOLID_FILL,2);
   bar(46,50,284,100);
   setcolor(WHITE);
   settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
   outtextxy(115,60,"ADVERTENCIA");
   setcolor(BLACK);
```

```
   outtextxy(70,80,"LA MEMORIA ESTA AGOTADA");
   rectangle(48,52,282,98);
   escapa( );
  }
 if(1!=fread(info, sizeof(struct alumno), 1, fp)) break;
 almacenar_dco(info, &principio, &final);
 }
 fclose(fp);
}

/*GUARDAR INFORMACION EN LA BASE DE DATOS */
void guardar (void)
 {
 struct alumno *info;
 FILE *fp;
 fp=fopen("alumnos.dat","wb");
 if(!fp){
 exit(1);
 }
 info=principio;
 while(info) {
   fwrite(info, sizeof(struct alumno), 1, fp);
   info=info→siguiente;
   }
 fclose(fp);
}

/* ASIGNAR LA CALIFICACION OBTENIDA POR EL ALUMNO */
calcular(char caletra[ ], int calnum)
{
 switch(calnum)
 {
 case 10: strcpy(caletra,"MB");
          break;
 case 9: strcpy(caletra,"MB");
          break;
 case 8: strcpy(caletra,"B");
          break;
 case 7: strcpy(caletra,"B");
          break;
 case 6: strcpy(caletra,"S");
          break;
 case 5: strcpy(caletra,"S");
          break;
 default: strcpy(caletra,"NA");
          break;
 }
return *caletra;
}

/* DESPLEGAR UN MENSAJE DE VALIDACION DEL TEMA O SUBTEMA VISTO ANTERIORMENTE */
ventana(int respsn)
{
   char *sn=" ", *si="S", *no="N";
   abre( );
   setbkcolor(MAGENTA);
   setfillstyle(SOLID_FILL,WHITE);
   bar(46,46,284,107);
   setcolor(2);
   settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
```

```
    outtextxy(68,55,"EL TEMA FUE VISTO POR TI");
    outtextxy(112,65,"ANTERIORMENTE");
    outtextxy(58,75,"DESEAS REPETIRLO DE NUEVO?");
    outtextxy(130,90,"SI   NO");
    rectangle(48,48,282,105);
    rectangle(126,85,146,102);
    rectangle(175,85,195,102);
o.i=bioskey (1);
o.i=bioskey (0);
*sn=toupper(o.c[0]);
if((*sn!='si) && (*sn!='no)){
     do
          {
          sound(1240);
          delay(100),
          nosound( );
          o.i=bioskey (1);
          o.i=bioskey (0);
          *sn=toupper(o.c[0]);
          }
 while((*sn!='si) && (*sn!='no));
 }
 if(*sn=='si)
    {
    respsn=1;
    }
 if(*sn=='no)
    {
    respsn=0;
    }
 cierra( );
 return respsn;
}

/* DESPLEGAR UN MENSAJE SI EXISTE ALGUN ERROR Y/O FALTA ARCHIVOS */
notema(char hist[])
{
    abre( );
    setfillstyle(SOLID_FILL,BLUE);
    bar(46,46,287,117);
    setcolor(BLACK);
    settextstyle(DEFAULT_FONT,HORIZ_DIR,1);
    outtextxy(55,55,"ERROR DE EJECUCION FALTA EL");
    outtextxy(55,65,"ARCHIVO →");
    line(120,68,165,68);
    setcolor(2);
    outtextxy(175,65,hist);
     setcolor(BLACK);
    outtextxy(56,75,"ES NECESARIO QUE INTERVENGA");
    outtextxy(122,85,"EL PROFESOR");
    outtextxy(68,95,"PRESIONA CUALQUIER TECLA");
    outtextxy(112,105,"PARA CONTINUAR");
    rectangle(48,48,285,115);
    getch( );
    cierra( );
}
```

Anexo B

Manual de Usuario.

MANUAL DEL USUARIO

LOGMAT: SISTEMA TUTORIAL DE LOGICA MATEMATICA

El sistema tutorial LOGMAT esta dirigido a profesores y alumnos que intervienen en el proceso de enseñanza-aprendizaje de la asignatura de Matemáticas I; este tutorial se constituye a partir de dos subsistemas: Subsistema de Control de Alumnos y Subsistema de Instrucción.

El Subsistema de Control de Alumnos ha sido desarrollado para ser utilizado exclusivamente por el profesor, debido a que es él, el encargado de accesar la información que le permitirá al alumno tener acceso a la utilización del Subsistema de Instrucción.

El Subsistema de Instrucción es la parte medular del sistema tutorial y ha sido diseñado para que el alumno que cursa la asignatura de lógica matemática lo ejecute.

Temas y subtemas que son abordables dentro del Subsistema de Instrucción.

1. Introducción a la lógica matemática.
   - 1.1 Definición de lógica matemática.
   - 1.2 Métodos lógicos
   - 1.3 Lenguaje simbólico.

2. Proposiciones.
   - 2.1 Definición de proposiciones
   - 2.2 Simples y compuestas
   - 2.3 Conectivos lógicos y tablas de verdad
     - 2.3.1 Definición de conectivos lógicos
     - 2.3.2 Definición de tablas de verdad
     - 2.3.3 Negación y doble negación
     - 2.3.4 Conjunción
     - 2.3.5 Disyunción
     - 2.3.6 Implicación condicional
     - 2.3.7 Doble condicional
   - 2.4 Tipos de proposiciones
     - 2.4.1 Tautológicas
     - 2.4.2 Contradictorias
     - 2.4.3 Contingentes
   - 2.5 Verdad formal y verdad empírica.

3. Argumentos
   - 3.1 Composición de un argumento
   - 3.2 Validez lógica de un argumento
   - 3.3 Leyes de implicación
     - 3.3.1 Modus ponendo ponens
     - 3.3.2 Modus tollendo tollens
     - 3.3.3 Modus tollendo ponens
     - 3.3.4 Silogismo hipotético
     - 3.3.5 Ley de la simplificación
     - 3.3.6 Ley de la conjunción
     - 3.3.7 Ley de la adición
   - 3.4 Leyes de equivalencia
   - 3.5 Demostración formal de la validez de un argumento

Es responsabilidad del profesor interactuar inicialmente con el Subsistema de Control de Alumnos; ya que como se mencionó anteriormente los datos que el profesor introduzca de cada uno de los alumnos del grupo de trabajo les permitirá el acceso al Subsistema de Instrucción.

OPERACION

Antes de iniciar a trabajar con el sistema tutorial tomaremos un poco de tiempo para mencionar que existen algunas instrucciones y utilización de teclas que se sugiere conveniente definir ya que tener conocimiento de ellas nos servirá de apoyo durante la utilización del sistema tutorial.

Salvar un archivo

En algunas ocasiones al emplear la computadora, si la apagamos antes de que "guardemos" el archivo que contiene toda la información generada, la perderemos y será imposible recuperarla. Salvar un archivo, es indicarle a la computadora que guarde la información en el disco para que posteriormente esta sea accesada y pueda ser consultada o modificada.

Recuperar o cargar un archivo

Generalmente vamos a tener algún archivo con información generada en el disco de la computadora, cuando se quiera consultar o trabajar en ella, se dice que estamos recuperando o cargando un archivo con información.

UTILIZACION DE ALGUNAS TECLAS

A continuación se describen algunas teclas cuya utilización será empleada dentro del sistema tutorial.

Tecla [ENTER], [RETURN], [INTRO] o [<=]

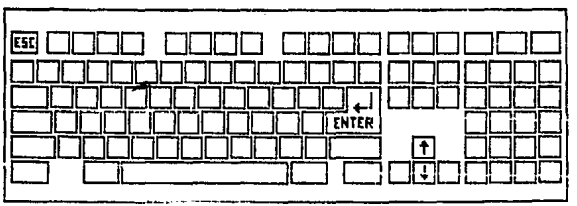
Esta tecla es utilizada para indicarle a la computadora que se concluyó de introducir una línea o letra.

Tecla [ESC]

Su utilización variara en función del subsistema en el que se esté trabajando. En el Subsistema de Control de Alumnos esta tecla nos servirá para indicar la salida de un menú subordinado hacia otro menú; en el Subsistema de Instrucción se utilizará para indicarle a la computadora la terminación de un tema o subtema.

Teclas de navegación.

Este conjunto de teclas nos permite movernos libremente dentro de un documento, pero, en el caso del sistema tutorial solamente se emplearán la tecla ↑ y la ↓ como una alternativa para elegir alguna de las opciones que se tienen en los menús del Subsistema de Control de Alumnos.



COMO INICIAR EL SISTEMA TUTORIAL DE INTRODUCCION A LA LOGICA MATEMATICA

SUBSISTEMA DE CONTROL DE ALUMNOS

Para poder iniciar el Subsistema de Control de Alumnos, accesar al sistema tutorial en el directorio o subdirectorio correspondiente y teclear **ALUMNOS** [Enter].

Después de unos segundos en la pantalla se podrá observar la pantalla de presentación del Subsistema de Control de Alumnos; se deberá pulsar la tecla [ESC] para salir del subsistema o cualquier otra tecla para continuar con la ejecución del mismo.

En el caso de optar para continuar con la utilización de subsistema después de unos segundos en la pantalla se observará el menú principal siguiente:

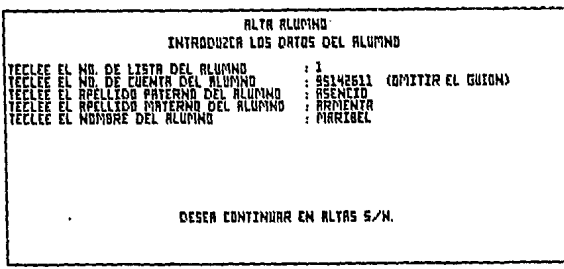
SELECCIONAR OPCIONES

Pasos:

- Para seleccionar la opción deseada, posicionarse con las teclas de navegación ↑ o ↓ sobre ella y pulsar la tecla [ENTER] o introducir el número de la opción deseada.
- Si se desea salir al sistema operativo y abandonar el menú principal entonces pulsar la tecla [ESC].

ALTAS

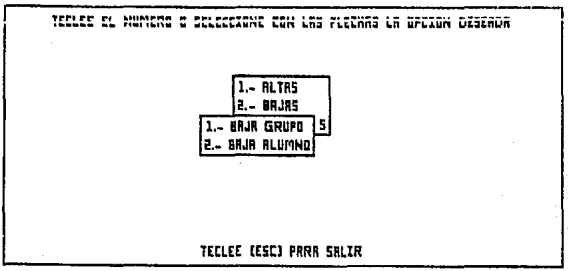
En esta opción se deberán introducir los datos que integraran la información de cada alumno con objeto de que posteriormente le sea permitido tener acceso al Subsistema de Instrucción; la elección de esta opción muestra la siguiente pantalla:



El Subsistema de Control de Alumnos accesará el mismo formato de pantalla, presentando al final un mensaje preguntando si se desea continuar dentro del proceso altas accesando datos para cada uno de los alumnos; al menos que no desee continuar con el proceso deberá pulsar la tecla "N", o "n". Al finalizar la introducción de los datos el subsistema presentará en pantalla un mensaje indicando que será necesario esperar un instante para que los datos accesados hasta ese momento, sean almacenados en el disco por el subsistema.

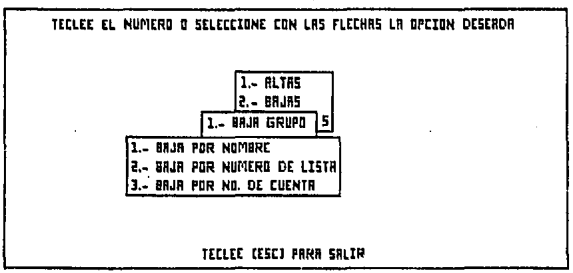
BAJAS

Esta opción realiza la tarea de eliminar información de un alumno o de un grupo; y al ser ella seleccionada nos muestra el siguiente menú:



Baja por Grupo Elimina la información de todos los alumnos que conforman un grupo. Si no existen alumnos dados de alta el subsistema desplegará un mensaje indicativo.

Baja por Alumno Elimina la información de un alumno; de acuerdo a un criterio de elección valiéndose para ello de mostrar la siguiente pantalla:



Si el alumno en cuestión no esta dado de alta el subsistema desplegará un mensaje indicativo. En ambas opciones el subsistema de control despliega una pantalla que cuestionará la verificación para proceder a eliminar información de un alumno o un grupo; si se opta por cancelar esta opción el subsistema presentará una pantalla que indique que la opción de baja ha sido suspendida.

CONSULTAS

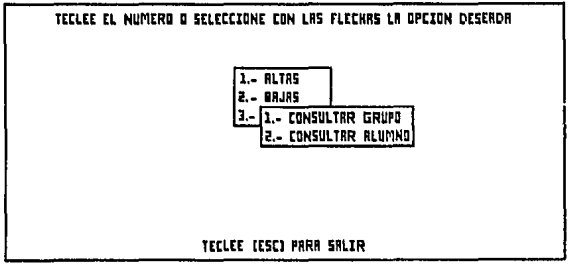
Esta opción permite realizar la consulta de la información de un alumno o un grupo, proporciona también información referente a temas, subtemas y su respectiva calificación.

El criterio de evaluación que se utilizará para determinar el aprovechamiento del alumno es el que la Legislación Universitaria en el artículo 4o. del capítulo primero del reglamento general de exámenes, señala: [1]

MB	(Muy bien)	igual a 10
B	(Bien)	igual a 8
S	(Suficiente)	igual a 6
NA	(No acreditada).	Carece de equivalencia numérica.
NP	(No presentado).	Carece de equivalencia numérica.

Dentro del sistema tutorial aquellos temas o subtemas que el alumno no haya estudiado serán señalados con "NP" (NO PRESENTADO).

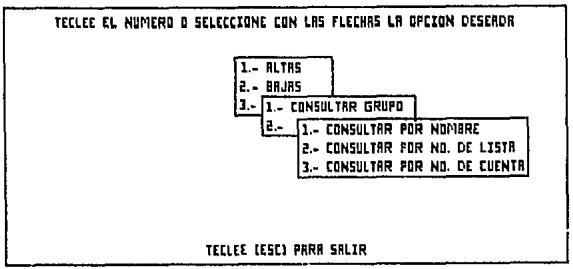
Al ser elegida esta opción será mostrado el siguiente menú:



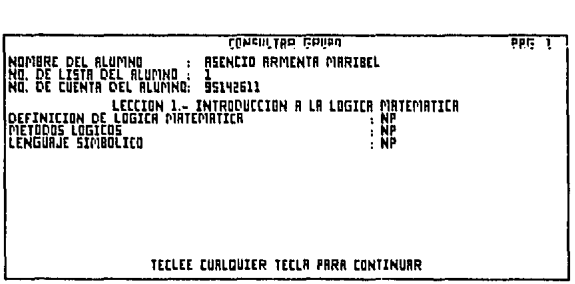
1 Legislación Universitaria, UNAM, Oficina del Abogado General, Dirección General de Estudios de Legislación Universitaria, 1992, p. 173.

Consulta por Grupo Despliega el resultado de la evaluación obtenida por el alumno durante su intervención con el Subsistema de Instrucción. Si no existen alumnos dados de alta el Subsistema desplegará un mensaje indicativo.

Consulta por alumno Despliega el resultado de la evaluación obtenida por el alumno en cada tema o subtema; de acuerdo a un criterio de elección, valiéndose para ello de mostrar la siguiente pantalla:



Cabe señalar que si el alumno no se encuentra dado de alta, el subsistema desplegará un mensaje indicativo lo cual significará que no existe información referente a ese alumno que pueda consultarse. Las pantallas que las opciones de consulta poseen para presentar información de un alumno o un grupo son las siguientes:



```
NOMBRE DEL ALUMNO      : ASENCIO ARMENTA MARIBEL                    PAG. 2
NO. DE LISTA DEL ALUMNO : 1
NO. DE CUENTA DEL ALUMNO: 95142611
                        LECCION 2.- PROPOSICIONES
DEFINICION DE PROPOSICIONES                          : NP
SIMPLES Y COMPUESTAS                                 : NP
CONECTIVOS LOGICOS Y TABLAS DE VERDAD                : NP
        DEFINICION DE CONECTIVOS LOGICOS             : NP
        DEFINICION DE TABLAS DE VERDAD               : NP
        NEGACION Y DOBLE NEGACION                    : NP
        CONJUNCION                                   : NP
        DISYUNCION                                   : NP
        IMPLICACION CONDICIONAL                      : NP
        DOBLE CONDICIONAL                            : NP
TIPOS DE PROPOSICIONES
        TAUTOLOGICAS                                 : NP
        CONTRADICTORIAS                              : NP
        CONTINGENTES (INDETERMINADAS)                : NP
VERDAD FORMAL Y VERDAD EMPIRICA                      : NP
```

```
NOMBRE DEL ALUMNO      : ASENCIO ARMENTA MARIBEL                    PAG. 3
NO. DE LISTA DEL ALUMNO : 1
NO. DE CUENTA DEL ALUMNO: 95142611
                        LECCION 3.- ARGUMENTOS
COMPOSICION DE UN ARGUMENTO                          : NP
VALIDEZ LOGICA DE UN ARGUMENTO                       : NP
LEYES DE IMPLICACION
        MODUS PONENDO PONENS                         : NP
        MODUS TOLLENDO TOLLENS                       : NP
        MODUS TOLLENDO PONENS                        : NP
        SILOGISMO HIPOTETICO                         : NP
        LEY DE LA SIMPLIFICACION                     : NP
        LEY DE LA CONJUNCION                         : NP
        LEY DE LA ADICION                            : NP
LEYES DE EQUIVALENCIA                                : NP
DEMOSTRACION FORMAL DE LA VALIDEZ DE UN ARGUMENTO    : NP
```

SUBSISTEMA DE INSTRUCCIÓN

Para poder iniciar el Subsistema de Instrucción, se deberá accesar al Sistema Tutorial en el directorio o subdirectorio correspondiente y teclear **LOGMAT** [ENTER]

Después de unos segundos en la pantalla se observará la presentación del Subsistema de Instrucción en la que se podrá pulsar la tecla [ESC] para salir o la tecla [ENTER] para continuar.

Si se quiere continuar entonces podrán observarse en la pantalla las siguientes instrucciones las cuales le permitirán al alumno lograr el máximo aprovechamiento del Subsistema de Instrucción.

1. Se recomienda que las sesiones de trabajo no excedan de una hora Transcurrido el tiempo sonará una melodía. Deberás salir del sistema y tomar un descanso de al menos 2 veces el tiempo de exposición del tema visto.

2. No es conveniente interrumpir el estudio en cualquier punto, sino al finalizar un tema o subtema.

3. El material educativo está organizado en secuencias de pantallas que tratan un tema, o parte de un tema, transcurrida la información el sistema presentará algunos ejercicios que reforzarán los conocimientos adquiridos.

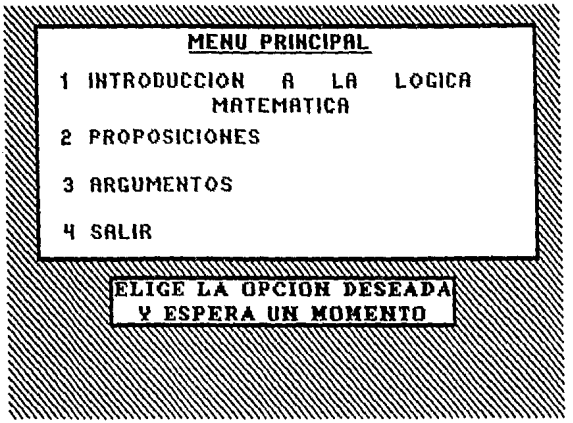
4. Al interactuar con el tutorial no te apresures Lee cuidadosamente

5. Nunca avances en el material ni respondas a un ejercicio antes de leer totalmente la pantalla.

6. Los ejercicios son de opción múltiple y en ellos tendrás 2 o 3 oportunidades de responder correctamente

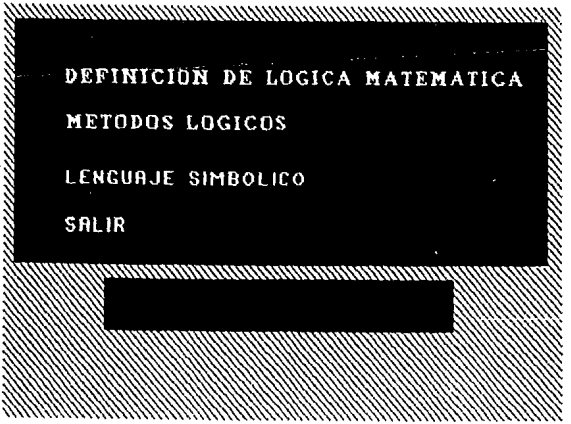
7 En el caso de que un ejercicio requiera más de una respuesta, si cometes un error la respuesta que hayas dado en las otras opciones se anulará aunque sean correctas. Vuélvelas a introducir.

Si se desea continuar con la utilización del subsistema después de unos segundos aparecerá una pantalla en la cual el alumno deberá introducir su nombre para que el subsistema le permita el acceso y pueda continuar; si al alumno le es denegado el acceso, el subsistema desplegará una pantalla señalando la restricción. De serle permitido el acceso al alumno, aparecerá en pantalla el siguiente menú principal:



El menú principal muestra las opciones de las tres lecciones que conforman la estructura temática de la asignatura de Matemáticas I. Es recomendable seguir el orden sistemático de cada una de las lecciones a fin de evitar omitir algún tema o subtema.

El tema **1, Introducción a la lógica matemática** del menú principal nos conducirá a un menú subordinado en el cual se podrán observar cuales son los temas que se encuentran relacionados a esa lección, como se observa a continuación:

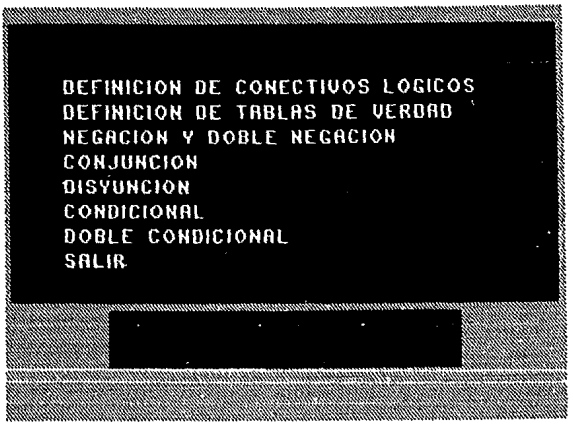


Es pertinente señalar que todos los menús que se utilizan en el Subsistema de Instrucción cuentan con una opción de salida [SALIR] que permiten abandonar al menú subordinado que se encuentra activo en ese momento en la pantalla permitiendo así regresar hacia el menú inmediato anterior permitiendo la salida a través de menús hasta lograr llegar al sistema operativo.

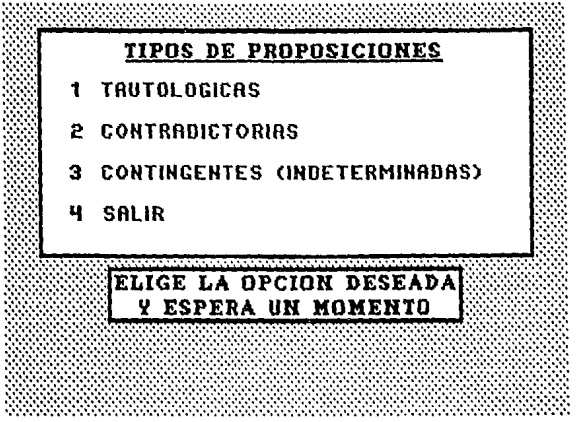
El tema 2, Proposiciones del menú principal, nos conducirá a otro menú donde se encuentran los temas y subtemas relacionados a ese tema y nos mostrará respectivamente los menús subordinados siguientes:

Los temas 3 y 4 de este menú cuentan con subtemas asociados y, al ser elegida alguna de sus opciones nos mostrarán los siguientes menús subordinados:

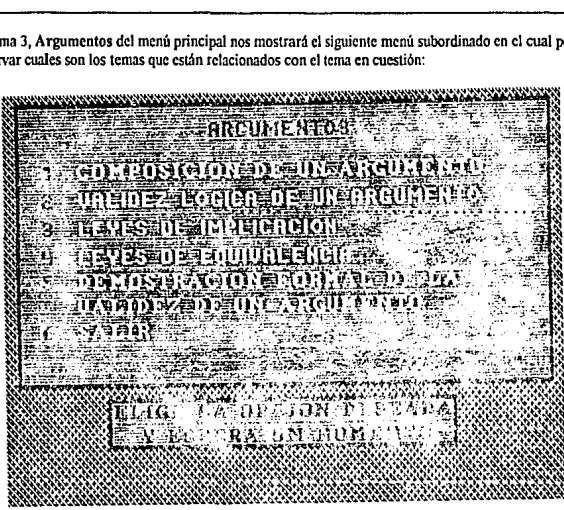
Menú del tema 3 Conectivos lógicos y tablas de verdad.



Menú del tema 4 Tipos de proposiciones.



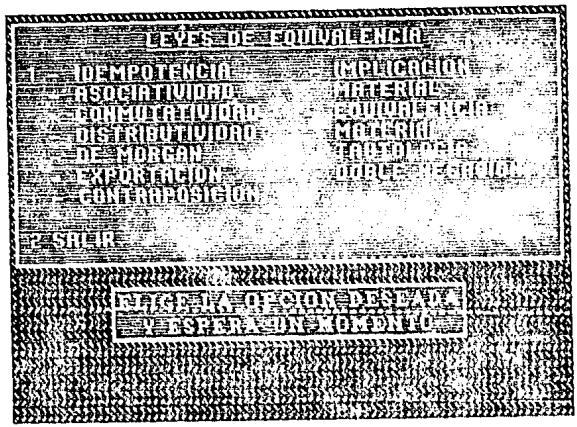
El tema 3, Argumentos del menú principal nos mostrará el siguiente menú subordinado en el cual podremos observar cuales son los temas que están relacionados con el tema en cuestión:



En este menú subordinado existen subtemas asociados a los temas 3 y 4, al seleccionar alguna de sus opciones nos proporcionará respectivamente los siguientes menús:

Menú del tema 3 Leyes de implicación.

Menú del tema 4 Leyes de equivalencia.



Cabe señalar que cada tema o subtema tiene asociado uno o más segmentos de información y uno o más segmentos de ejercicios, los cuales deberán ser respondidos mediante opción múltiple por el alumno para su inmediata evaluación. El subsistema realizará el conteo de aciertos y errores que son introducidos durante la interacción con los ejercicios los cuales son dados a conocer por el subsistema, en el momento de concluir la ejecución de algún tema o subtema. Del total de aciertos y errores es obtenida la resultante de aplicar una regla de tres dando como resultado un valor numérico que será comparado con un rango de valores internos que permitirán definir la calificación otorgada al alumno por el sistema tutorial. La calificación obtenida por el alumno será almacenada en el archivo que posee el Subsistema de Control en su correspondiente ubicación y al lado derecho del tema o subtema visto

Aunque se recomienda no interrumpir el estudio en cualquier punto del Subsistema de Instrucción; se podrá concluir un tema o subtema pulsando la tecla [ESC] para indicar que se quiere terminar con la sesión de trabajo; esto implicará que el alumno no será evaluado dándose por hecho de que el tema o subtema aun no ha sido visto, por lo tanto seguirá señalado dentro del Subsistema de Control de Alumnos con "NP" (No presentado)

Anexo C

Ficha de Evaluación.

FICHA DE EVALUACION

Estimado profesor:

La presente ficha tiene como objetivo evaluar el sistema tutorial.

Instrucciones para el llenado de la ficha de evaluación.

1. Realice la lectura completa del manual.
2. Ejecute todo el sistema tutorial antes de llenar la ficha.
3. Ejecute el sistema tutorial por partes, intentando identificar las características pedagógicas y operacionales.
4. Evalúe al sistema tutorial de acuerdo con la escala numérica presentada en esta ficha.
5. Señale su juicio para cada ítem de la Parte II de la ficha considerando una escala de 1 a 5, donde el 1 represente la evaluación más negativa y el 5 la más positiva.

   1 2 3 4 5

6. Comente en la Parte III su apreciación como evaluador; siéntase en libertad de expresar su opinión.

Fecha de evaluación:

____/ ____/ 19____

Parte I:

¿ Existe manual de operación ?

( ) si ( ) no

¿ Fue de utilidad el manual de operación ?

( ) si ( ) no

¿ El sistema tutorial es autoexplicativo?

( ) si ( ) no

¿ El sistema tutorial exige un lugar especial para ser utilizado ?

( ) si ( ) no

Si la respuesta es afirmativa. ¿ Que lugar?

( ) laboratorio
( ) salón de clases
( ) otro

¿ El sistema tutorial exhibe diferentes grados de dificultad en las cuestiones presentadas ?

( ) si ( ) no

¿ Los ejercicios son generados aleatoriamente?

( ) si ( ) no

¿ Los alumnos necesitan explicación adicional para utilizar el sistema tutorial?

( ) si ( ) no

¿ El sistema tutorial ofrece ayuda o "pistas" al alumno?

( ) si ( ) no

¿ El alumno puede alterar la secuencia del sistema tutorial?

( ) si ( ) no

¿ El sistema tutorial acepta abreviaturas ?

( ) si ( ) no

¿ Son adecuados los recursos utilizados?

( ) si ( ) no

¿ Utiliza el sistema tutorial sugerencias didácticas innovadoras ?

( ) si ( ) no

¿ Que objetivo educacional enfatiza más el sistema tutorial?

( ) descripción
( ) contrastación
( ) análisis

Parte II:

¿ Que tan adecuadas son las instrucciones para la operación del sistema tutorial ?

1 2 3 4 5

¿ Son claramente explicados los comandos con los que el sistema tutorial realiza sus funciones ?

1 2 3 4 5

¿ Es adecuada la presentación del sistema tutorial en cuanto a colocación de títulos, figuras, textos y comandos?

1 2 3 4 5

¿ La cantidad de información en la pantalla facilita su lectura ?

1 2 3 4 5

¿ Son atrayentes las pantallas del sistema tutorial ?

1 2 3 4 5

¿ Son adecuados los conceptos para ser transmitidos por el sistema tutorial ?

1 2 3 4 5

¿ Favorece la lectura y comprensión del tutorial el ritmo de presentación de las pantallas ?

1 2 3 4 5

¿Es oportuno el señalamiento que realiza el tutorial al detectar errores de evaluación ?

1 2 3 4 5

¿ Proporciona el sistema tutorial diferentes tipos de mensajes de error para señalar las equivocaciones ?

1 2 3 4 5

¿ Son congruentes los objetivos educacionales con el contenido temático del sistema tutorial ?

1 2 3 4 5

¿ Se puede suspender e iniciar el sistema tutorial, determinando así su secuencia ?

1 2 3 4 5

¿ El resultado total obtenido durante la realizacion de los ejercicios es dado a conocer oportunamente por el sistema tutorial ?

1 2 3 4 5

¿ Ofrece el sistema tutorial una secuencia explicativa para las respuestas equivocadas ?

1 2 3 4 5

¿ Son coherentes las ideas presentadas con la fundamentación didáctica adoptada por el profesor ?

1 2 3 4 5

¿ Posee el sistema tutorial elementos motivadores para estimular a los alumnos ?

1 2 3 4 5

¿ El texto presentado en el sistema tutorial tiene continuidad y secuencia ?

1 2 3 4 5

¿ Permite el sistema tutorial que se puedan intentar varias respuestas durante la realización de los ejercicio ?

1 2 3 4 5

¿ El sistema tutorial se torna activo al requerir de la participación del usuario ?

1 2 3 4 5

Parte III. Comentario del evalnador.

INDICE DE FIGURAS.

BIBLIOGRAFIA.

Influencia de los Sistemas Modernos de Información en el Desarrollo Educativo. (Memoria 4a. Conferencia)
México, UNAM, 1988, 392 p.

Legislación Universitaria.
México, UNAM, Oficina del Abogado General, Dirección General de Estudios de Legislación Universitaria, 1993, 320. p.

Plan de Estudio 1993 (Tomo I).
México, UNAM/ FCA/McGraw-Hill, 1993, 380 p.

Alatorre, Roberto.
Lógica.
(13a. ed.), México, Porrúa, 1983, 347 p.

Arnaz, José Antonio.
Introducción a la Lógica Simbólica.
(3a. ed.), México, Trillas, 1989, 103 p.

Asley, Ruth.
Matemáticas Fundamentales para Computación.
México, Limusa, 1988, 305 p.

Bork, Alfred.
El Ordenador en la Enseñanza (Análisis y Perspectivas de Futuro).
España, Gustavo Gili, 1986, 268 p.

Calderón, Enrique.
Computadoras en la Educación.
México, Trillas, 1988, 258 p.

Cole K, S.
Mathematical Logic.
(5a. ed),E.U.A, Jonh Wiley and Sons, 89 p.

Copi, M. Irving.
Lógica Simbólica.
(6a. ed.), México, CECSA, 1987, 407 p.

Davidoff, L. Linda.
Introducción a la Psicología.
(2a. ed.), México, McGraw-Hill, 1984, 777 p.

Ellis, A. B.
The Use and Misuse of Computers in Education.
Nueva York, McGraw-Hill, 1974, 226 p.

Ezzell, Ben
Programación de Gráficos en Turbo C ++.
E.U.A., Addison-Wesley Iberoamericana, 1993, 587 p.

Fernández, M.
Enseñanza Asistida por Ordenador.
España, Anaya, 1983, 184 p.

Flores, Conrado.
Nociones de Lógica Matemática.
México, Trillas, 1986, 139 p

Fuentes, Arturo.
Diagnóstico: Fundamentos, Metodología y Técnicas.
Seminario y Taller de Metodología.
UNAM, Facultad de Ingeniería, División de Estudios de Posgrado, Departamento de Ingeniería de Sistemas.
México, (3a. Impresión),1992, 87 p.

García, Enrique.
Técnicas Modernas en la Educación.
México, Trillas, 1971, 102 p.

Giordanio, E. y Edelstein, R.
La Creación de Programas Didácticos (Lenguajes y Sistemas de Autor).
España, Gustavo Gili, 1987, 152 p.

Greenfield P. M.
Mind and Media: Effects of Television, Computers and Video Games.
E.U.A., Fontana, 1984, 193 p.

Jamsa, Kris.
Lenguaje C (Biblioteca de Funciones).
México, McGraw-Hill, 1987, 283 p.

Jasso, Pedro.
Lógica Matemática.
México, McGraw-Hill, 1989, 67 p.

Marques, P. y Saucho, J.
Como Introducir y Utilizar el Ordenador en la Clase.
España, CEAC, 1987, 144 p.

Papert, S.
Mindstorms: Children, Computers and Powerful Ideas.
E.U.A, Brighton Harvest Press, 1980, 230 p.

Pressman, S. Roger.
Ingeniería del Software (Un Enfoque Práctico).
(3a. ed.), México, McGraw-Hill, 1993, 807 p.

Sánchez Guerrero, Gabriel.
Evaluación de Programas Sociales: Un Enfoque a Programas Académicos Universitarios.
México, UNAM, Facultad de Ingeniería, División de Estudios de Posgrado,1994, 191 p.

Schildt, Herbert.
Programación en Turbo C.
España, McGraw-Hill, 1988, 373 p.

Schildt, Herbert.
Advanced C.
(2a. ed.), E.U.A., Borland Osborne / McGraw-Hill, 1989, 403 p.

Schildt, Herbert .
The Art of C.
E.U.A., Borland Osborne / McGraw-Hill, 1991, 453 p.

Schildt, Herbert.
Turbo C: The Complete Reference.
E.U.A., Borland Orborne / McGraw-Hill, 1988, 891 p.

Stroustrup, Bjarne.
El Lenguaje de Programación C++.
E.U.A., Addison-Wesley Iberoamericana, 1993, 710 p.

Suppes, Patrick.
Introducción a la Lógica Simbólica.
(2a. ed.), México, CECSA, s/año, 40 p.

Zubieta Russi, Gonzalo.
Apuntes de Análisis Lógico.
México, UNAM, 1992, 111p.

HEMEROGRAFIA.

Carbajal, E. y Pesina, C.
"La Computación en la Enseñanza"
Ciencia y Desarrollo.
México, CONACyT.
Vol. XV, No. 88, Sep.- Oct., 1989, 160 p.

Bernardino de Campos, G. y Elliot Ligia G..
"Evaluación de Productos Educativos"
Tecnología y Comunicación Educativas.
México, ILCE.
Año 6, No. 18 Julio 1991, 60 p.

Lauterbach Roland y Frey Karl.
"Los Programas de Informática para la Enseñanza: Balance y Perspectivas".
Perspectivas.
Francia, UNESCO.
Vol. 17, No. 3, 1987, (63).