Directorio de Profesores del Curso: LENGUAJE DE PROGRAMACION BASIC CON APLICACIONES FEBRERO-MARZO DE 1983

ING. HERIBERTO OLGUIN ROMO (COORDINADOR)
Director de Captura y Proceso
Dirección General de Política Informática
Instituto Nacional de Estadística
Geografía e Informática
Secretaría de Programación y Presupuesto
Arcos de Belen No. 2 P.H.
México, D.F.
761 62 27

M. en C. RICARDO CIRIA MERCE
Subdirector de Nuevos Proyectos
Coordinación de la Administración Escolar
Edificio IIMAS, P.B.; Cubículo F.
U N A M
México, D.F.
550 50 46 y 550 50 45

ING. HECTOR JAVIER ARRONA URREA
Subdirector de Producción
Dirección de Captura y Proceso
Dirección General de Política Informática
Instituto Nacional de Geografía, Estadística
e Informatica de la
S E P
Arcos de Belén No. 2 P.B.
México, D.F.
761 62 27

M. EN C. CARLOS AUGUSTO RAMOS LARIOS
Jefe del Departamento de Información
Dirección de Proveeduría
Av. Revolución No. 2040-1°Piso
UNAM
550 52 15 Ext. 4027 y 548 97 75

ING. JUAN ALEJANDRO JIMENEZ GARCIA
COORDINADOR DE INVESTIGACION Y DESARROLLO
CENTRO DE CALCULO
FACULTAD DE INGENIERIA
UNAM
550 52 15 Ext. 3734

ING. JORGE ONTIVEROS JUNCO
Coordinador de Investigación y Desarrollo
Centro de Cálculo de la Facultad de Ingeniería
UNAM
550 52 15 Ext. 3734

LENGUAJE DE PROGRAMACION BASIC

(PRIMERA PARTE)

INDICE DE TEMAS

DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA
UNAM

FEBRERO DE 1983.

LENGUAJE DE PROGRAMACION BASIC

(PRIMERA PARTE)

D I S T R I B U C I O N D E T I E M P O S

Horarios:

Viernes: 17:00 a 21:00 Hrs. y Sábados: 9:00 a 14:00 Hrs.

Primer Viernes. (18 de febrero)

Lugar: Palacio de Minería

Descansos: 18:15 a 18:30 y 19:45 a 20:00 Hrs.

1. Introducción al PED.

   Dos o tres aplicaciones

   Sesión práctica de motivación

   Juegos

   Ricardo Cicia Marco
   Heriberto Diguín Romo

Primer Sábado. (19 de Febrero)

Lugar: Centro de Cálculo de la Fac. Ing. C. U.

Descansos: 10:30 a 10:45 y 12:15 a 12:30

3. Elementos de BASIC.

   Ricardo Cicia Marco
   Héctor Javier Arrona Urrea

4. Instrucciones BASIC:

   LET, INPUT, PRINT, END, GO TO

   Ejemplos en pizarrón.

Segundo Viernes. (25 de febrero)

Lugar: Palacio de Minería

Descansos: 18:15 a 18:30 y 19:45 a 20:00

4. Continúa

READ, DATA, RESTORE, REM, STOP, IF

Ejemplos de pizarrón.

Héctor Javier Arrona Urrea
Jorge Ontiveros Junco

Segundo Sábado. (26 de febrero)

Lugar: CECAFI

Descansos: 10:30 a 10:45 y 12:15 a 12:30

5. Introducción a la TRS-80 III (todo).

Jorge Ontiveros Junco
Carlos Ramos Larios

4. Continúa

ON, GO TO, FOR, NEXT

Funciones intrínsecas.

Práctica.

Tercer Viernes. (4 de marzo)

Lugar: Palacio de Minería

Elementos de Programación Estructurada.

Carlos Ramos Larios
Alejandro Jiménez García

Tercer Sábado. (5 de marzo)

Lugar: CECAFI

Práctica de Programación Estructurada.

8. Formatos.

Práctica de formatos.

Alejandro Jiménez García
Héctor Javier Arrona Urrea

Cuarto Viernes. (11 de marzo)

Lugar: Palacio de Minería

Variables con subíndice.

Funciones de caracteres.

Ricardo Cirio Horca
Heriberto Olguín Romo

Cuarto Sábado. (12 de marzo)

Lugar: CECAFI

Práctica de V(I) y F de C.

Subprogramas.

Práctica de subprogramas.

Héctor Javier Arrona Urrea
Jorge Ontiveros Junco

Quinto Viernes. (18 de marzo)

Lugar: Palacio de Minería

Dos aplicaciones prácticas
- Análisis
- Diseño
- Programación
- Producción

Ricardo Ciria Merce
Carlos Ramos Larios

Quinto Sábado. (19 de marzo)

Lugar: CECAFI

- Práctica de aplicaciones.
- Pantalla.
- Clausura.

Heriberto Olguín Romo
Carlos Ramos Larios

LENGUAJE DE PROGRAMACION BASIC - PRIMERA PARTE

ESTAS NOTAS FUERON ELABORADAS POR LOS

EXPOSITORES:

ING.HECTOR JAVIER ARRONA URREA
M.EN C.JOSE RICARDO CIRIA MERCE
ING.JUAN ALEJANDRO JIMENEZ GARCIA
ING. HERIBERTO OLGUIN ROMO
ING. JORGE ONTIVEROS JUNCO
M.EN C.CARLOS AUGUSTO RAMOS LARIOS

Y SE ANEXAN UNOS ARTICULOS QUE APOYAN
AL CURSO.

LENGUAJE DE PROGRAMACION BASIC

(PRIMERA PARTE)

INDICE DE TEMAS

DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA
UNAM

FEBRERO DE 1983.

LENGUAJE DE PROGRAMACION BASIC

(PRIMERA PARTE)

D I S T R I B U C I O N D E T I E M P O S

Horarios:

Viernes: 17:00 a 21:00 Hrs. y Sábados: 9:00 a 14:00 Hrs.

Primer Viernes. (18 de febrero)

Lugar: Palacio de Minería
Descansos: 18:15 a 18:30 y 19:45 a 20:00 Hrs.

1. Introducción al PED.

Dos o tres aplicaciones

Sesión práctica de motivación

Juegos

Ricardo Ciria Merce
Heriberto Olguín Romo

Primer Sábado. (19 de febrero)

Lugar: Centro de Cálculo de la Fac. Ing. C. U.

Descansos: 10:30 a 10:45 y 12:15 a 12:30

3. Elementos de BASIC.

Ricardo Ciria Merce
Héctor Javier Arrona Urrea

4. Instrucciones BASIC:

LET, INPUT, PRINT, END, GO TO

Ejemplos en pizarrón.

Segundo Viernes. (25 de febrero)

Lugar: Palacio de Minería

Descansos: 18:15 a 18:30 y 19:45 a 20:00

4. Continúa

READ, DATA, RESTORE, REM, STOP, IF

Ejemplos de pizarrón.

Héctor Javier Arrona Urrea
Jorge Ontiveros Junco

Segundo Sábado. (26 de febrero)

Lugar: CECAFI

Descansos: 10:30 a 10:45 y 12:15 a 12:30

5. Introducción a la TRS-80 III (todo).

Jorge Ontiveros Junco
Carlos Ramos Larios

4. Continúa

ON, GO TO, FOR, NEXT

Funciones intrínsecas.

Práctica.

Tercer Viernes. (4 de marzo)

Lugar: Palacio de Minería

Elementos de Programación Estructurada.

Carlos Ramos Larios
Alejandro Jiménez García

Tercer Sábado. (5 de marzo)

Lugar: CECAFI

Práctica de Programación Estructurada.

8. Formatos.

Práctica de formatos.

Alejandro Jiménez García
Héctor Javier Arrona Urrea

Cuarto Viernes. (11 de marzo)

Lugar: Palacio de Minería

Variables con subíndice.

Funciones de caracteres.

Ricardo Ciria Merce
Heriberto Olguín Romo

Cuarto Sábado. (12 de marzo)

Lugar: CECAFI

Práctica de V(I) y F de C.

Subprogramas.

Práctica de subprogramas.

Héctor Javier Arrona Urrea
Jorge Ontiveros Junco

Quinto Viernes. (18 de marzo)

Lugar: Palacio de Minería

Dos aplicaciones prácticas

Análisis
Diseño
Programación
Producción

Ricardo Cirla Merce
Carlos Ramos Larios

Quinto Sábado. (19 de marzo)

Lugar: CECAFI

Práctica de aplicaciones.

Pantalla.

Clausura.

Heriberto Olguín Romo
Carlos Ramos Larios

LENGUAJE DE PROGRAMACION BASIC

(PRIMERA PARTE)

INDICE DE TEMAS

DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA
UNAM

FEBRERO DE 1983.

LENGUAJE DE PROGRAMACION BASIC

(PRIMERA PARTE)

D I S T R I B U C I O N D E T I E M P O S

Horarios:

Viernes: 17:00 a 21:00 Hrs. y Sábados: 9:00 a 14:00 Hrs.

Primer Viernes. (18 de febrero)

Lugar: Palacio de Minería

Descansos: 18:15 a 18:30 y 19:45 a 20:00 Hrs.

1. Introducción al PED.

   Dos o tres aplicaciones

   Sesión práctica de motivación

   Juegos

   Ricardo Cirio Merce
   Heriberto Olguín Romo

Primer Sábado. (19 de febrero)

Lugar: Centro de Cálculo de la Fac. Ing. C. U.

Descansos: 10:30 a 10:45 y 12:15 a 12:30

3. Elementos de BASIC.

   Ricardo Cirio Merce
   Héctor Javier Arrona Urrea

4. Instrucciones BASIC:

   LET, INPUT, PRINT, END, GO TO

   Ejemplos en pizarrón.

Segundo Viernes. (25 de febrero)

Lugar: Palacio de Minería

Descansos: 18:15 a 18:30 y 19:45 a 20:00

4. Continúa

READ, DATA, RESTORE, REM, STOP, IF

Ejemplos de pizarrón.

Héctor Javier Arrona Urrea
Jorge Ontiveros Junco

Segundo Sábado. (26 de febrero)

Lugar: CECAFI

Descansos: 10:30 a 10:45 y 12:15 a 12:30

5. Introducción a la TRS-80 III (todo).

Jorge Ontiveros Junco
Carlos Ramos Larios

4. Continúa

ON, GO TO, FOR, NEXT

Funciones intrínsecas.

Práctica.

Tercer Viernes. (4 de marzo)

Lugar: Palacio de Minería

Elementos de Programación Estructurada.

Carlos Ramos Larios
Alejandro Jiménez García

Tercer Sábado. (5 de marzo)

Lugar: CECAFI

Práctica de Programación Estructurada.

8. Formatos.

Práctica de formatos.

Alejandro Jiménez García
Héctor Javier Arrona Urrea

Cuarto Viernes. (11 de marzo)

Lugar: Palacio de Minería

Variables con subíndice.

Funciones de caracteres.

Ricardo Ciria Merce
Heriberto Olguín Romo

Cuarto Sábado. (12 de marzo)

Lugar: CECAFI

Práctica de V(I) y F de C.

Subprogramas.

Práctica de subprogramas.

Héctor Javier Arrona Urrea
Jorge Ontiveros Junco

Quinto Viernes. (18 de marzo)

Lugar: Palacio de Minería

Dos aplicaciones prácticas
- Análisis
- Diseño
- Programación
- Producción

Ricardo Ciria Merce
Carlos Ramos Larios

Quinto Sábado. (19 de marzo)

Lugar: CECAFI

Práctica de aplicaciones.

Pantalla.

Clausura.

Heriberto Olguín Romo
Carlos Ramos Larios

LENGUAJE DE PROGRAMACION BASIC - CON APLICACIONES
(PRIMERA PARTE)

LA PROXIMA GENERACION DE COMPUTADORAS

FEBRERO, 1983

LA PROXIMA GENERACION DE COMPUTADORAS

Enrique Calderón Alzati

(Comunicaciones de la Fundación Rosenblueth, Enero de 1982)

1. INTRODUCCION.

Aparecidas en los inicios de la década cincuenta, las computadoras constituyen hoy en día, uno de los instrumentos que mayor influencia ejerce en todas las formas de actividad humana.

El estudio de su evolución, orientado a la prospección de nuevas tecnologías y formas de aplicación, constituyen actualmente motivo de esfuerzos y dedicación de gobiernos, industrias e institutos de investigación.

En ese estudio se conjugan, tanto los aspectos tecnológicos (que incluyen la micro-electrónica y la física del estado sólido), como los matemáticos y los filosóficos (incluyendo la lógica, la lingüística y la teoría de la recursión) y finalmente los aspectos relacionados con los procesos cognitivos y adaptativos del hombre y otros organismos vivientes. Todos ellos habrán de contribuir a los próximos desarrollos de la computación.

En el presente reporte se pretende examinar en forma integral algunos de estos aspectos, para fundamentar la tesis concerniente al papel que ha tenido el concepto de "lo que es la computación" como motor del cambio y enriquecimiento de la tecnología de cómputo.

Con este reporte queremos iniciar una serie de documentos orientados a mostrar la gran riqueza conceptual que existe en la computación, la cual es totalmente desconocida en nuestro país, debido a la mala orientación de los programas educativos y al escaso interés y poca difusión de estos temas.

Cuatro Esquemas de la Computación.

Si bien las ideas de máquinas inteligentes, robots y automatización industrial, estuvieron presentes en la primera mitad del siglo XX, las primeras computadoras son creadas para servir como instrumentos de cálculo en los institutos de investigación, organismos militares y estadísticos y departamentos de las grandes corporaciones industriales.

Aunque los resultados logrados eran importantes, el mercado para los grandes y costosos equipos de cálculo, era necesariamente restringido, motivando a las industrias de cómputo a la búsqueda de nuevas aplicaciones.

El uso de las computadoras en la automatización de procesos administrativos, que si bien requerían cálculos relativamente sencillos, hacían necesaria la ejecución de grandes volúmenes de proceso por su naturaleza repetitiva, se convirtió pronto en la principal área de aplicación.

En esta nueva forma de aplicación conocida como "proceso electrónico de datos", los requerimientos principales se encontraban en la entrada y salida de grandes volúmenes de información.

La utilización masiva y problemática de tarjetas perforadas como forma principal de almacenamiento y transferencia de información, orientó los esfuerzos industriales y de investigación al desarrollo de nuevos medios de almacenamiento.

La aparición de la cinta magnética y el desarrollo de las impresoras de alta velocidad (del orden de 1,000 LPM) constituyeron los elementos principales del éxito logrado en el "proceso electrónico de datos" que abrió el mercado de las computadoras a los sectores financiero, industrial y gobierno, todos ellos con enormes problemas de administración.

El siguiente paso conceptual importante, en el proceso de diversificación y desarrollo de la computación fue la aparición de los llamados "sistemas de información" que tuvieron gran éxito durante la década de los setenta.

Entre los avances tecnológicos que hicieron factible este nuevo avance, podemos citar la introducción del disco magnético (por IBM en 1960), el desarrollo de multiprogramación y la capacidad de utilización de teletipos y posteriormente de terminales interactivas para dar lugar al "tiempo compartido" introducido simultáneamente en forma comercial por Burroughs, Univac y General Electric.

Aspectos típicos de los sistemas de información, es el uso de un acervo central de datos, organizado de acuerdo con un esquema preconcebido, que permita la consulta simultánea de la información, por parte de varios usuarios.

Aunque los sistemas de información contienen generalmente mecanismos de actualización permanente para el acervo de información; esta función es realizada en forma centralizada, a través de un canal único establecido como parte de la infraestructura del organismo usuario.

Mientras que en el "proceso electrónico de datos" el énfasis estaba en la automatización de los procesos administrativos de las instituciones, con el enfoque de los "sistemas de información" el énfasis se daba en el estudio de la organización misma, orientada a la construcción de modelos sobre los cuales, la información quedaba estructurada como una imagen evolutiva y adaptiva de la realidad.

En esta forma los resultados que previamente se obtenían mediante el proceso electrónico de datos, podían lograrse como meros productos secundarios de los sistemas de información.

Aunque las aplicaciones más conocidas de los sistemas de información se dieron en los Bancos y Compañías de Aviación, su impacto en organismos gubernamentales, industrias y corporaciones comerciales fue también considerable.

Un punto más que es necesario mencionar en relación a los sistemas de información, es el referente a la evolución de sus aplicaciones que se dieron inicialmente a nivel operativo (sistemas de cuentas corrientes en bancos y de reservaciones aéreas), después a los niveles administrativos intermedios y finalmente a la planeación y la toma de decisiones de alto nivel.

Un concepto bien conocido y utilizado, es el que se refiere al computador como un amplificador de la capacidad intelectual del hombre.

En el uso de sistemas de información estaba implícita una nueva forma de aplicación destinada a la amplificación de una de nuestras capacidades fundamentales, la de comunicación, que constituye la esencia misma de las sociedades humanas.

Su importancia sólo puede ser comprendida, al observar la relación que existe entre comunicación y desarrollo, entre la palabra escrita y el florecimiento de las primeras civilizaciones, entre la imprenta y la historia moderna.

Los sistemas de comunicación por computadora resultan cuando la capacidad de alimentar y actualizar los datos de un acervo es otorgado a todos (algunos) los usuarios del sistema de información. "

Los sistemas de comunicación por computadora, representan un avance cualitativo sobre otras formas de comunicación en cuanto que:

a) La distribución de mensajes es selectiva y asociativa (se envía sólo a los receptores que la requieren o que cumplen ciertas condiciones).

b) La recepción de mensajes es también selectiva (sólo se aceptan mensajes sobre temas determinados por el receptor, o provenientes de fuentes también selectivas).

c) La capacidad de distribución y recepción es tanto instantánea, como independiente del tiempo, es decir que la información queda disponible para cuando sea necesaria.

Un claro ejemplo de sistemas de comunicación se da nuevamente en los sistemas bancarios y de reservaciones aéreas, cuando sus operadores tienen la facultad de actualizar el acervo con los montos pagados o depositados en un caso y las reservaciones o cancelaciones realizadas en el otro.

la conceptualización de los sistemas de información, al igual que el proceso de datos como casos particulares de "procesos de comunicación" es evidente.

Como en las etapas anteriores, el desarrollo de los sistemas de comunicación abren las puertas a nuevas formas de aplicación ya predecibles, estando su éxito sustentado en nuevos avances tecnológicos entre los que es conveniente destacar:

- Los desarrollos de la micro electrónica conocidos como LSI y VLSI (Large System Integration y Very Large Systems Intregration), que lograron entre otras cosas reducir y permitir la utilización masiva de equipos.
- El desarrollo de las telecomunicaciones que permitieron conectar equipos de cómputo distantes y abrieron la posibilidad de transferir grandes volúmenes de información.
- El desarrollo de las redes de cómputo basadas en las posibilidades de comunicación "inteligente" entre equipos de cómputo, que constituyeron la infraestructura principal de los procesos de comunicación.

En resumen, hemos analizado cuatro formas o esquemas de aplicación de la computación, que a nuestro modo de ver constituyen la esencia de las cuatro generaciones de computadoras a las que se hace referencia en la literatura técnica y comercial de cómputo y que se asocia más a los aspectos puramente físicos de los componentes utilizados en cada una de ellas.

Nuestra tesis delineada con mayor detalle en (1), pretende establecer que siendo los avances de la electrónica importantes, no constituyen sino uno de los factores de la evolución tecnológica del cómputo.

Por otra parte, aunque puede parecer que el desarrollo conceptual de la computación ha llegado a su fin, la realidad es totalmente distinta, nuevas generaciones de computadores con objetivos más amplios y de mayor trascendencia habrán de seguir en las próximas décadas. Un análisis al respecto ocupa la atención de las secciones siguientes de este reporte.

2. CUATRO GENERACIONES DE COMPUTADORAS.

Desde el punto de vista puramente tecnológico, la evolución de la computación es sorprendente y contempla otros aspectos adicionales a los de la electrónica y componentes utilizados; aspectos tales como el tipo de dispositivos periféricos y el software evolucionaron en forma paralela y acorde a las necesidades del mercado que crecía con las nuevas aplicaciones.

Aunque es difícil encasillar en un esquema de cuatro etapas el desarrollo casi continuo de los nuevos sistemas de computación, la utilización de ese esquema es importante por su relación a las formas de utilización ya mencionadas y por la posibilidad de discutir la evolución tecnológica en términos sencillos.

Algunas inexactitudes en los años asociados a la aparición de cada etapa, así como a las estructuras y dimensiones de los equipos, se deben a la existencia de algunas computadoras de transición adelantadas a su época y a la tecnología existente, como lo fue la B-5500 de Burroughs (entre otros equipos).

Una vez aclarado este punto haremos una descripción breve de los aspectos más relevantes de cada una de las generaciones.

a) Computadoras de la primera generación:

Entrada al mercado:	1950 aproximadamente.
Aplicación principal:	Instumentos de cálculo.
Tecnología utilizada:	Tubos de vacío.
	Memoria de cilindro magnético.
Unidades periféricas:	Lectoras y perforadoras de tarjetas y cinta de papel, equipo unitario, etc.
Sistema operativo:	No existía.
Lenguajes de programación:	Lenguaje de máquina, ensambladores primitivos.
Alfabeto:	Numérico.
Administración:	Trivial, no se requería.
Aspectos cuantitativos:	M. Central 1,000 a 8,000 palabras.
	Proceso 10^4 ops/seg.
	Precio 100,000 a 2,5 millones US.
Modelos típicos:	IBM-650 Bendix-G15, Univac 5590, Bull-P1, IBM-709.

b) Computadoras de la segunda generación:

Entrada al mercado:	1960 aproximadamente.
Aplicaciones principales:	Proceso de datos.
	Instrumento de cálculo.
Tecnología utilizada:	Transistores y ferritas.
Unidades periféricas:	Lectoras y perforadoras de tarjetas, impresoras y cintas magnéticas.

Sistema operativo:	Rudimentario, controla periféricos, inicia y termina tareas.
Lenguajes de programación:	Ensambladores y primeros compiladores (FORTRAN, ALGOL).
Alfabéto:	Números y letras, algunos caracteres especiales.
Facilidades adicionales:	Existencia de bibliotecas.
Administración:	Primitiva, planeación de producción con procesos masivos.
Aspectos cuantitativos:	MC 8,000 a 32,000 palabras. Procesadores 10^5 ops/seg. Precio 10^5 a 10^6 US.
Modelos típicos:	CDC-160, IBM -7090, IBM-1401, Burroughs-5500, RCA-305, Bendix G20, CDC-3600.

c) Computadoras en la tercera generación:

Entrada al mercado:	Aproximadamente entre 1968 y 1970.
Aplicaciones principales:	Sistemas de información.
Tecnología utilizada:	Circuitos integrados (LSI), memoria de películas magnéticas.
Unidades periféricas:	Cintas y discos magnéticos, terminales de video y teletipos.
Arquitectura:	Multiprogramación, multiproceso, sistemas de interrupción, optimización de código.
Lenguajes y facilidades de programación:	Lenguajes de alto nivel COBOL, PL, Bases de Datos (DMS).
Alfabeto:	Números, letras y caracteres especiales.

Sistema operativo:	Manejo de archivos, multiproceso, memoria dinámica, memoria virtual, etc.
Facilidades adicionales:	Edición y prueba interactiva de programas.
Administración:	Compleja y especializada.
Aspectos cuantitativos:	MC 64 a 256 K palabras. Procesador 10^6 ops/seg. Memoria secundaria 10^9 caracteres. Rango de precios $5X10^4$ a 10^7 US.
Modelos típicos:	IBM-360, Burroughs-6700, PDP 10, PDP 11, Univac-1106, Cyber 170.

d) Computadoras de la cuarta generación:

Entrada al mercado:	Entre 1977 y 1981.
Aplicaciones principales:	Sistemas de comunicación, sistemas de información para negocios pequeños, uso personal.
Tecnologías utilizadas:	Micro-electrónica VLSI, memorias-Nos (meta) oxide sylicates).
Unidades periféricas:	Terminales inteligentes, discos y cintas magnéticas, equipos de graficación, lectores ópticos y digitalizadores.
Arquitectura:	Proceso distribuido, uso de microprocesadores.
Lenguajes y facilidades de programación:	Bases de datos distribuidas, lenguajes interactivos, descriptivos y gráficos.
Alfabeto:	Irrestricto, mayúsculas y minúsculas, símbolos matemáticos, alfabeto Arabe, Japonés, etc.

Sistema operativo:	Proceso sin interrupción, comunicación entre máquinas, rutinas de recuperación, etc.
Facilidades adicionales:	Metaprocesadores, correo electrónico, manejadores de texto.
Administración:	Muy simple para equipos personales. Muy complejo para redes de proceso distribuido.
Aspectos cuantitativos:	Memoria Central 64K a 10^7 caracteres. Procesadores 10^7 ops/seg. Memoria secundaria 10^{10} caracteres. Rango de precios 10^3 a 10^8 US.
Modelos típicos Grandes:	IBM-4330, Univac 1100, Burroughs B-6900, 7900.
Medianos:	Prime 550 HP 3100 VAC5.
Pequeños:	Apple, TRS80

3. LAS EXPECTATIVAS NO CUMPLIDAS Y LAS NUEVAS APLICACIONES.

La existencia de robots y máquinas industriales automáticas, concebidas en la primera mitad del siglo XX antes que las primeras computadoras entraran en acción, no ha sido totalmente plasmada en la medida que el avance tecnológico permitiera esperar.

Las razones de este retraso son múltiples e incluso factores sociales relacionados con el posible desempleo que estas nuevas máquinas podrían causar, por la falta de alternativas y legislatura adecuada.

Entre los factores tecnológicos para la introducción de las máquinas automáticas, estuvieron las abultadas dimensiones, escasa confiabilidad y las restricciones que su uso imponía sobre el ambiente de operación, que difícilmente pueden darse en las instalaciones industriales.

Por otra parte, la idea original de las máquinas automáticas capaces de realizar operaciones totalmente repetitivas, sin ninguna capacidad adaptiva ni característica inteligente, tuvo que ser desechada por inoperante para dar lugar a la idea de máquinas inteligentes capaces de realizar rutinas alternas y adaptarse a las necesidades que la producción de objetos distintos implica. Para todo ello, la capacidad perceptiva de estos robots resultaba indispensable.

Con el desarrollo de la micro-electrónica de la nueva tecnología de las comunicaciones y del reconocimiento de formas, logrados en la cuarta generación de computadoras, el camino para los robots y la automatización industrial a escala masiva, ha quedado abierto. Los desarrollos actuales de la inteligencia artificial tendrán un impacto importante en este proceso y a su vez se nutrirán del mismo.

Otro problema aún no resuelto satisfactoriamente, es el relacionado con la traducción mecánica de lenguajes; aquí el problema es de otra naturaleza porque no obstante la experiencia ancestral que el hombre tiene en esta actividad, jamás se había preocupado por la mecánica de este proceso, por lo que ha sido necesario el desarrollo previo de modelos lingüísticos formales, para hacer posible la creación de los más simples programas de traducción; los avances logrados hasta hoy y los esfuerzos que actualmente se hacen en esta dirección, permiten prever su realización en los próximos años, con un impacto social formidable.

Finalmente, la gran capacidad de almacenamiento y recuperación de información lograda con las computadoras actuales, no compite con la capacidad del hombre que a través de sus sentidos (especialmente el visual), está captando constantemente imágenes que implican volúmenes impresionantes de información, que son de inmediato sintetizadas en forma óptima

para su almacenamiento. Este hecho ligado indiscutiblemente a la estructura y funcionamiento esencialmente distinto al de las computadoras actuales, permiten concluir el desarrollo futuro de otras arquitecturas de cómputo.

4. ALGUNAS TENDENCIAS EVOLUTIVAS.

Aunque los aspectos señalados en las secciones anteriores muestran un desarrollo dramático y señalan algunas tendencias importantes, otras no son explícitas y requieren un análisis adicional.

a) Interacción hombre-máquina.

El acercamiento de las computadoras al hombre, ha representado un reto continuo, a investigadores, profesionistas e industriales de la computación; los avances logrados son impresionantes.

Así, mientras las computadoras de la primera generación, fueron instrumentos exclusivos de sus creadores y de algunos cuantos científicos, que requerían de meses de estudio y un alto grado de conocimientos previos de matemáticas, para construir programas relativamente simples, las máquinas de la segunda generación principiaron a estar al alcance de ingenieros y técnicos, gracias a la aparición de FORTRAN y de otros lenguajes similares, así como a la incorporación de los sistemas operativos, que habrían de relevar al programador de las tareas relacionadas con el control de las unidades periféricas.

Con la introducción de la terminal de consulta, del tiempo compartido y de nuevos lenguajes conceptualmente más simples, no sólo las tareas de programación se simplificaron, sino que

por primera vez, el personal no técnico (cajeros y empleados administrativos) pudieran entrar en contacto con la computadora, dejando atrás los tiempos en que las transacciones y los documentos eran enviados a los "centros de cálculo" o "unidades informáticas" para su proceso por medios misteriosos y fuera del control del usuario.

En la etapa de la cuarta generación, el proceso de acercamiento continuó, ahora las secretarias hacen uso de terminales para redactar oficios y cartas, los jefes tienden a depender cada vez más para comunicarse con sus empleados y consultar bancos de datos financieros, los empleados de hoteles e industrias hacen igualmente uso de ellas y en las escuelas, se convierten en un excelente apoyo para la experimentación y la enseñanza, el nivel de acercamiento es tal que aún los niños hacen uso cotidiano de ellas.

Factores de este fenómeno que habrá de intensificarse en el futuro, son la reducción de los costos, la confiabilidad y reducción de requerimientos para su operación y la concepción misma de los equipos de cómputo como intrumentos de uso masivo. El desarrollo de nuevos lenguajes interactivos y gráficos ha constituido un factor adicional de este proceso.

De este análisis es posible concluir que la interacción hombre/máquina habrá de continuar su crecimiento en las próximas generaciones de computadoras, buscando nuevas formas de acercamiento que hoy en día principian a delinearse.

b) La computación no numérica.

En sus orígenes, la computación fue totalmente numérica, no sólo en lo que respecta a la naturaleza de los datos y tipos de problemas que se pretendía resolver, sino al lenguaje mismo de programación.*

La dotación de caracteres alfabéticos a los mecanismos de impresión y el estudio de la lingüística (Chomsky, Bar-Hillel, etc.) hizo factible el uso de lenguajes no numéricos para la programación, y la dependencia cada vez menor de los números, sin embargo, las aplicaciones fundamentales de la computación siguieron ligadas a los aspectos numéricos, durante las dos generaciones siguientes de computadoras.

Esta afirmación puede parecer extraña al lector, acostumbrado a pensar en aplicaciones administrativas en las que se hace referencia a nombres de personas, de empresas, de productos, o de lugares geográficos. Aunque esto es totalmente cierto,

* En las primeras computadoras, las instrucciones eran codificadas en forma totalmente numérica, de acuerdo con el modelo introducido por John Von Neumann, quién utilizó en su concepción de la computadora, las ideas expuestas por Kurt Godel que asociaba un número (programaba cada función computable).

es también válido mencionar que en todos estos casos la información alfabética nunca constituye la parte esencial del proceso; así por ejemplo, el nombre del empleado es sólo una etiqueta que se anexa al resultado del cálculo de percepciones y deducciones, en la generación de un documento de pago, y si bien es cierto que para la sociedad, quien importa es el portador del documento, para el computador el trabajo central reside en las cifras.

Esta observación no implica que el uso de la computadora fuese exclusivamente en aplicaciones numéricas, pues a partir de 1960, diversos esfuerzos de investigación se centraron en problemas no numéricos, logrando resultados que han tenido y tendrán una influencia notable en el desarrollo futuro de la computación.

Como ya se ha mencionado, el origen de las investigaciones sobre problemas no numéricos se situó en la lingüística formal, con la búsqueda de algoritmos que hicieran factible la traducción mecánica de lenguajes. Posteriormente, el desarrollo de LISP y otros lenguajes como Snobol y L^6, hicieron factibles los trabajos en inteligencia artificial y manipulación de símbolos algebráicos, estableciendo la posibilidad de dotar a las máquinas con capacidad de deducción para resolver problemas de lógica.

Es así que para los inicios de la década de los setenta existiesen máquinas capaces de jugar ajedrez, reconocer objetos a partir de su descripción lingüística (para lo cual el computador requiere de la conexión de un equipo de televisión) y de hacer deducciones lógicas. Unos años más tarde las primeras máquinas capaces de sostener conversaciones sencillas, eran estudiados en algunos institutos de investigación.

La construcción de máquinas con capacidad de inferencia y conversación generalizada, y la creación de "sistemas de conocimiento" como intrumento de uso masivo, constituye hoy en día el objetivo a lograr para la siguiente generación de computadoras.*

Otro proceso de desarrollo menos complejo, ha tenido lugar durante los últimos años en el campo de la computación no numérica y constituye hoy en día, la confirmación del éxito que habrán de tener los sistemas futuros de comunicación y conocimientos. Se trata de los populares "procesadores de palabra" que son utilizados en forma masiva, para la preparación de cartas y documentos técnicos.

* Figenbaum Conference on Fifth Generation Computer Systems. Tokio, 1981.

Con el procesador de palabras el usuario tiene acceso a un conjunto de instrucciones que le permiten editar textos, corregir errores y dar la presentación y organización más adecuada. El envío y almacenamiento de documentos con costos extraordinariamente reducidos, hace posible la utilización masiva de esta tecnología, que para los inicios de la década 80, ha principiado a substituir a la máquina de escribir y a los demás equipos tradicionales de oficina.

Sin lugar a dudas este proceso de orientación de la computación a las aplicaciones no numéricas represente una tendencia irreversible para el futuro.

c) La orientación de las investigaciones.

Como en otras áreas de la tecnología moderna, el período requerido para que los resultados exitosos de una investigación sean aplicados masivamente, se reduce, a unos cuantos meses o a lo más un par de años; es así que, el análisis de la evolución de los "objetos de estudio" relacionados con la computación, constituye una tercer área de análisis por su importancia.

A este respecto podemos decir que mientras en la primera generación, la preocupación y motivos de investigación, se orientaban al desarrollo de los métodos de cálculo numérico y a la búsqueda de nuevos dispositivos de almacenamiento magnético,

durante la siguiente etapa los temas de estudio estuvieron centrados en la lingüística, la utilización de transistores y circuitos integrados, y en la naturaleza de los procesos administrativos.·

Al iniciarse la era de la tercera generación de computadoras, el interés de la investigación varió hacia el análisis del funcionamiento de las organizaciones (con el enfoque de sistemas), al uso de la tecnología LSI, al desarrollo de los sistemas de información y hacia la inteligencia artificial.

Finalmente, los temas de interés al entrar al mercado la cuarta generación, eran el estudio de la sociedad y las interacciones que en su seno tenían los organismos industriales y financieros y por otro lado las nuevas formas de aplicación de la micro-electrónica, y el estudio de las comunicaciones digitales.

Actualmente y para el futuro, el tema principal de estudio ha cambiado nuevamente para centrarse en "el hombre", en los procesos cognitivos, de la inferencia y la acumulación de conocimientos. Los avances de la micro-electrónica hacen factible imitar los procesos adaptivos de los organismos biológicos y la posibilidad de construir mosaicos de pequeños autómatas para formar "tejidos" cuya capacidad de asimilación y análisis de imágenes, sea similar a la del ojo humano, reconocido ya como uno de los sistemas más eficientes de recolección de información.

5. LAS COMPUTADORAS DE LA QUINTA GENERACION.

Con estas nuevas orientaciones y con un mercado de dimensiones jamás superadas en su magnitud y diversificación, se principian a definir en los centros de investigación norteamericanos y europeos, los conceptos que darán lugar a la quinta generación de computadoras.*

Paralelamente y quizás con un empuje mayor, la industria japonesa desarrolla un proyecto de gran magnitud para poner en operación una nueva generación de computadoras substancialmente distintas a las existentes, en los primeros años de la década de 1990.*

Estas nuevas máquinas habrán de caracterizarse por la utilización de enjambres procesadores microscópicos, operando simultáneamente para recibir y clasificar información, por su capacidad básica de inferencia y generación de "conocimientos" y esquemas generales, a partir de información particular, así como por su estrecha relación con el hombre.

* Este proyecto fue presentado por primera ocasión en la Conferencia sobre "Computadora de la Quinta Generación", realizada en octubre de 1981, mencionada previamente en este reporte.

El sistema de control habrá de seguir y utilizar principios ya conocidos hoy en día, sólo se usan en experimentos de inteligencia artificial, como los relacionados con las máquinas LISP y de Flujo de datos (Data flow machines).

El desarrollo de la computación no numérica como forma principal de aplicación, y la interacción más intensa entre hombre y máquina estarán también en el centro del proceso. Finalmente, la importancia de la comunicación entre máquinas, y con ello, entre los hombres que la poseen, habrán de delinear en buena medida el futuro de la civilización.

6. NUESTRA PARTICIPACION EN EL PROCESO.

Cuál es y será nuestro papel en esta revolución ¿la de observadores pasivos del proceso? ¿Pasará inadvertido como otros tantos cambios tecnológicos de cuyas implicaciones políticas y sociales nos enteramos sólo años después? ¿Podríamos ser capaces de asimilar esta vez las tecnologías emergentes? ¿Qué planes podemos formular para tomar una parte activa por modesta que sea en este proceso de cambio? Quizás estas sean algunas preguntas que hoy deberíamos hacernos.

LA PROXIMA GENERACION DE COMPUTADORAS

Enrique Calderón Alzati

(Comunicaciones de la Fundación Rosenblueth, Enero de 1982)

1. INTRODUCCION.

Aparecidas en los inicios de la década cincuenta, las computadoras constituyen hoy en día, uno de los instrumentos que mayor influencia ejerce en todas las formas de actividad humana.

El estudio de su evolución, orientado a la prospección de nuevas tecnologías y formas de aplicación, constituyen actualmente motivo de esfuerzos y dedicación de gobiernos, industrias e institutos de investigación.

En ese estudio se conjugan, tanto los aspectos tecnológicos (que incluyen la micro-electrónica y la física del estado sólido), como los matemáticos y los filosóficos (incluyendo la lógica, la linguística y la teoría de la recursión) y finalmente los aspectos relacionados con los procesos cognitivos y adaptativos del hombre y otros organismos vivientes. Todos ellos habrán de contribuir a los próximos desarrollos de la computación.

En el presente reporte se pretende examinar en forma integral algunos de estos aspectos, para fundamentar la tesis concerniente al papel que ha tenido el concepto de "lo que es la computación" como motor del cambio y enriquecimiento de la tecnología de cómputo.

Con este reporte queremos iniciar una serie de documentos orientados a mostrar la gran riqueza conceptual que existe en la computación, la cual es totalmente desconocida en nuestro país, debido a la mala orientación de los programas educativos y al escaso interés y poca difusión de estos temas.

Cuatro Esquemas de la Computación.

Si bien las ideas de máquinas inteligentes, robots y automatización industrial, estuvieron presentes en la primera mitad del siglo XX, las primeras computadoras son creadas para servir como instrumentos de cálculo en los institutos de investigación, organismos militares y estadísticos y departamentos de las grandes corporaciones industriales.

Aunque los resultados logrados eran importantes, el mercado para los grandes y costosos equipos de cálculo, era necesariamente restringido, motivando a las industrias de cómputo a la búsqueda de nuevas aplicaciones.

El uso de las computadoras en la automatización de procesos administrativos, que si bien requerían cálculos relativamente sencillos, hacían necesaria la ejecución de grandes volúmenes de proceso por su naturaleza repetitiva, se convirtió pronto en la principal área de aplicación.

En esta nueva forma de aplicación conocida como "proceso electrónico de datos", los requerimientos principales se encontraban en la entrada y salida de grandes volúmenes de información.

La utilización masiva y problemática de tarjetas perforadas como forma principal de almacenamiento y transferencia de información, orientó los esfuerzos industriales y de investigación al desarrollo de nuevos medios de almacenamiento.

La aparición de la cinta magnética y el desarrollo de las impresoras de alta velocidad (del orden de 1,000 LPM) constituyeron los elementos principales del éxito logrado en el "proceso electrónico de datos" que abrió el mercado de las computadoras a los sectores financiero, industrial y gobierno, todos ellos con enormes problemas de administración.

El siguiente paso conceptual importante, en el proceso de diversificación y desarrollo de la computación fue la aparición de los llamados "sistemas de información" que tuvieron gran éxito durante la década de los setenta.

Entre los avances tecnológicos que hicieron factible este nuevo avance, podemos citar la introducción del disco magnético (por IBM en 1960), el desarrollo de multiprogramación y la capacidad de utilización de teletipos y poste-

riormente de terminales interactivas para dar lugar al "tiempo compartido" introducido simultáneamente en forma comercial por Burroughs, Univac y General Electric.

Aspectos típicos de los sistemas de información, es el uso de un acervo central de datos, organizado de acuerdo con un esquema preconcebido, que permita la consulta simultánea de la información, por parte de varios usuarios.

Aunque los sistemas de información contienen generalmente mecanismos de actualización permanente para el acervo de información, esta función es realizada en forma centralizada, a través de un canal único establecido como parte de la infraestructura del organismo usuario.

Mientras que en el "proceso electrónico de datos" el énfasis estaba en la automatización de los procesos administrativos de las instituciones, con el enfoque de los "sistemas de información" el énfasis se daba en el estudio de la organización misma, orientada a la construcción de modelos sobre los cuales, la información quedaba estructurada como una imagen evolutiva y adaptiva de la realidad.

En esta forma los resultados que previamente se obtenían mediante el proceso electrónico de datos, podían lograrse como meros productos secundarios de los sistemas de información.

Aunque las aplicaciones más conocidas de los sistemas de información se dieron en los Bancos y Compañías de Aviación, su impacto en organismos gubernamentales, industrias y corporaciones comerciales fue también considerable.

Un punto más que es necesario mencionar en relación a los sistemas de información, es el referente a la evolución de sus aplicaciones que se dieron inicialmente a nivel operativo (sistemas de cuentas corrientes en bancos y de reservaciones aéreas), después a los niveles administrativos intermedios y finalmente a la planeación y la toma de decisiones de alto nivel.

Un concepto bien conocido y utilizado, es el que se refiere al computador como un amplificador de la capacidad intelectual del hombre.

En el uso de sistemas de información estaba implícita una nueva forma de aplicación destinada a la amplificación de una de nuestras capacidades fundamentales, la de comunicación, que constituye la esencia misma de las sociedades humanas.

Su importancia sólo puede ser comprendida, al observar la relación que existe entre comunicación y desarrollo, entre la palabra escrita y el florecimiento de las primeras civilizaciones, entre la imprenta y la historia moderna.

Los sistemas de comunicación por computadora resultan cuando la capacidad de alimentar y actualizar los datos de un acervo es otorgado a todos (algunos) los usuarios del sistema de información.

Los sistemas de comunicación por computadora, representan un avance cualitativo sobre otras formas de comunicación en cuanto que:

a) La distribución de mensajes es selectiva y asociativa (se envía sólo a los receptores que la requieren o que cumplen ciertas condiciones).

b) La recepción de mensajes es también selectiva (sólo se aceptan mensajes sobre temas determinados por el receptor, o provenientes de fuentes también selectivas).

c) La capacidad de distribución y recepción es tanto instantánea, como independiente del tiempo, es decir que la información queda disponible para cuando sea necesaria.

Un claro ejemplo de sistemas de comunicación se da nuevamente en los sistemas bancarios y de reservaciones aéreas, cuando sus operadores tienen la facultad de actualizar el acervo con los montos pagados o depositados en un caso y las reservaciones o cancelaciones realizadas en el otro.

La conceptualización de los sistemas de información, al igual que el proceso de datos como casos particulares de "procesos de comunicación" es evidente.

Como en las etapas anteriores, el desarrollo de los sistemas de comunicación abren las puertas a nuevas formas de aplicación ya predecibles, estando su éxito sustentado en nuevos avances tecnológicos entre los que es conveniente destacar:

- Los desarrollos de la micro electrónica conocidos como LSI y VLSI (Large System Integration y Very Large Systems Intregration), que lograron entre otras cosas reducir y permitir la utilización masiva de equipos.

- El desarrollo de las telecomunicaciones que permitieron conectar equipos de cómputo distantes y abrieron la posibilidad de transferir grandes volúmenes de información.

- El desarrollo de las redes de cómputo basadas en las posibilidades de comunicación "inteligente" entre equipos de cómputo, que constituyeron la infraestructura principal de los procesos de comunicación.

En resumen, hemos analizado cuatro formas o esquemas de aplicación de la computación, que a nuestro modo de ver constituyen la esencia de las cuatro generaciones de computadoras a las que se hace referencia en la literatura técnica y comercial de cómputo y que se asocia más a los aspectos puramente físicos de los componentes utilizados en cada una de ellas.

Nuestra tesis delineada con mayor detalle en (1), pretende establecer que siendo los avances de la electrónica importantes, no constituyen sino uno de los factores de la evolución tecnológica del cómputo.

Por otra parte, aunque puede parecer que el desarrollo conceptual de la computación ha llegado a su fin, la realidad es totalmente distinta, nuevas generaciones de computadores con objetivos más amplios y de mayor trascendencia habrán de seguir en las próximas décadas. Un análisis al respecto ocupa la atención de las secciones siguientes de este reporte.

2. CUATRO GENERACIONES DE COMPUTADORAS.

Desde el punto de vista puramente tecnológico, la evolución de la computación es sorprendente y contempla otros aspectos adicionales a los de la electrónica y componentes utilizados; aspectos tales como el tipo de dispositivos periféricos y el software evolucionaron en forma paralela y acorde a las necesidades del mercado que crecía con las nuevas aplicaciones.

Aunque es difícil encasillar en un esquema de cuatro etapas el desarrollo casi continuo de los nuevos sistemas de computación, la utilización de ese esquema es importente por su relación a las formas de utilización ya mencionadas y por la posibilidad de discutir la evolución tecnológica en términos sencillos.

Algunas inexactitudes en los años asociados a la aparición de cada etapa, así como a las estructuras y dimensiones de los equipos, se deben a la existencia de algunas computadoras de transición adelantadas a su época y a la tecnología existente, como lo fue la B-5500 de Burroughs (entre otros equipos).

Una vez aclarado este punto haremos una descripción breve de los aspectos más relevantes de cada una de las generaciones.

a) Computadoras de la primera generación:

Entrada al mercado:	1950 aproximadamente.
Aplicación principal:	Instumentos de cálculo.
Tecnología utilizada:	Tubos de vacío.
	Memoria de cilindro magnético.
Unidades periféricas:	Lectoras y perforadoras de tarjetas y cinta de papel, equipo unitario, etc.
Sistema operativo:	No existía.
Lenguajes de programación:	Lenguaje de máquina, ensambladores primitivos.
Alfabeto:	Numérico.
Administración:	Trivial, no se requería.
Aspectos cuantitativos:	M. Central 1,000 a 8,000 palabras.
	Proceso 10^4 ops/seg.
	Precio 100,000 a 2,5 millones US.
Modelos típicos:	IBM-650 Bendix-G15, Univac SS90, Bull-PT, IBM-709.

b) Computadoras de la segunda generación:

Entrada al mercado:	1960 aproximadamente.
Aplicaciones principales:	Proceso de datos.
	Instrumento de cálculo.
Tecnología utilizada:	Transistores y ferritas.
Unidades periféricas:	Lectoras y perforadoras de tarjetas, impresoras y cintas magnéticas.

Sistema operativo:	Rudimentario, controla periféricos, inicia y termina tareas.
Lenguajes de programación:	Ensambladores y primeros compiladores (FORTRAN, ALGOL).
Alfabéto:	Números y letras, algunos caracteres especiales.
Facilidades adicionales:	Existencia de bibliotecas.
Administración:	Primitiva, planeación de producción con procesos masivos.
Aspectos cuantitativos:	MC 8,000 a 32,000 palabras. Procesadores 10^5 ops/seg.
	Precio 10^5 a 10^6 US.
Modelos típicos:	CDC-160, IBM -7090, IBM-1401, Burroughs-5500, RCA-305, Bendix G20, CDC-3600.

c) Computadoras en la tercera generación:

Entrada al mercado:	Aproximadamente entre 1968 y 1970.
Aplicaciones principales:	Sistemas de información.
Tecnología utilizada:	Circuitos integrados (LSI), memoria de películas magnéticas.
Unidades periféricas:	Cintas y discos magnéticos, terminales de video y teletipos.
Arquitectura:	Multiprogramación, multiproceso, sistemas de interrupción, optimización de código.
Lenguajes y facilidades de programación:	Lenguajes de alto nivel COBOL, PL, Bases de Datos (DMS).
Alfabeto:	Números, letras y caracteres especiales.

Sistema operativo:	Manejo de archivos, multiproceso, memoria dinámica, memoria virtual, etc.
Facilidades adicionales:	Edición y prueba interactiva de programas.
Administración:	Compleja y especializada.
Aspectos cuantitativos:	MC 64 a 256 K palabras. Procesador 10^6 ops/seg. Memoria secundaria 10^9 caracteres. Rango de precios $5X10^4$ a 10^8 US.
Modelos típicos:	IBM-360, Burroughs-6700, PDP 10, PDP 11, Univac-1106, Cyber 170.

d) Computadoras de la cuarta generación:

Entrada al mercado:	Entre 1977 y 1981.
Aplicaciones principales:	Sistemas de comunicación, sistemas de información para negocios pequeños, uso personal.
Tecnologías utilizadas:	Micro-electrónica VLSI, memorias-Mos (metal oxide sylicates).
Unidades periféricas:	Terminales inteligentes, discos y cintas magnéticas, equipos de graficación, lectores-ópticos y digitalizadores.
Arquitectura:	Proceso distribuido, uso de microprocesadores.
Lenguajes y facilidades de programación:	Bases de datos distribuidas, lenguajes interactivos, descriptivos y gráficos.
Alfabeto:	Irrestricto, mayúsculas y minúsculas, símbolos matemáticos, alfabeto Arabe, Japonés, etc.

Sistema operativo:	Proceso sin interrupción, comunicación entre máquinas, rutinas de recuperación, etc.
Facilidades adicionales:	Metaprocesadores, correo electrónico, manejadores de texto.
Administración:	Muy simple para equipos personales. Muy complejo para redes de proceso distribuido.
Aspectos cuantitativos:	Memoria Central 64K a 10^7 caracteres. Procesadores 10^7ops/seg. Memoria secundaria 10^{10} caracteres. Rango de precios 10^3 a 10^8 US.
Modelos típicos Grandes:	IBM-4330, Univac 1100, Burroughs B-6900, 7900.
Medianos:	Prime 550 MP 3100 VACS.
Pequeños:	Apple, TRS80

3. LAS EXPECTATIVAS NO CUMPLIDAS Y LAS NUEVAS APLICACIONES.

La existencia de robots y máquinas industriales automáticas, concebidas en la primera mitad del siglo XX antes que las primeras computadoras entraran en acción, no ha sido totalmente plasmada en la medida que el avance tecnológico permitiera esperar.

Las razones de este retraso son múltiples e incluso factores sociales relacionados con el posible desempleo que estas nuevas máquinas podrían causar, por la falta de alternativas y legislatura adecuada.

Entre los factores tecnológicos para la introducción de las máquinas automáticas, estuvieron las abultadas dimensiones, escasa confiabilidad y las restricciones que su uso imponía sobre el ambiente de operación, que difícilmente pueden darse en las instalaciones industriales.

Por otra parte, la idea original de las máquinas automáticas capaces de realizar operaciones totalmente repetitivas, sin ninguna capacidad adaptiva ni característica inteligente, tuvo que ser desechada por inoperante para dar lugar a la idea de máquinas inteligentes capaces de realizar rutinas alternas y adaptarse a las necesidades que la producción de objetos distintos implica. Para todo ello, la capacidad perceptiva de estos robots resultaba indispensable.

Con el desarrollo de la micro-electrónica de la nueva tecnología de las comunicaciones y del reconocimiento de formas, logrados en la cuarta generación de computadoras, el camino para los robots y la automatización industrial a escala masiva, ha quedado abierto. Los desarrollos actuales de la inteligencia artificial tendrán un impacto importante en este proceso y a su vez se nutrirán del mismo.

Otro problema aún no resuelto satisfactoriamente, es el relacionado con la traducción mecánica de lenguajes; aquí el problema es de otra naturaleza porque no obstante la experiencia ancestral que el hombre tiene en esta actividad, jamás se había preocupado por la mecánica de este proceso, por lo que ha sido necesario el desarrollo previo de modelos linguísticos formales, para hacer posible la creación de los más simples programas de traducción; los avances logrados hasta hoy y los esfuerzos que actualmente se hacen en esta dirección, permiten prever su realización en los próximos años, con un impacto social formidable.

Finalmente, la gran capacidad de almacenamiento y recuperación de información lograda con las computadoras actuales, no compite con la capacidad del hombre que a través de sus sentidos (especialmente el visual), está captando constantemente imágenes que implican volúmenes impresionantes de información, que son de inmediato sintetizadas en forma óptima

para su almacenamiento. Este hecho ligado indiscutiblemente a la estructura y funcionamiento esencialmente distinto al de las computadoras actuales, permiten concluir el desarrollo futuro de otras arquitecturas de cómputo.

4. ALGUNAS TENDENCIAS EVOLUTIVAS.

Aunque los aspectos señalados en las secciones anteriores muestran un desarrollo dramático y señalan algunas tendencias importantes, otras no son explícitas y requieren un análisis adicional.

a) Interacción hombre-máquina.

El acercamiento de las computadoras al hombre, ha representado un reto continuo, a investigadores, profesionistas e industriales de la computación; los avances logrados son impresionantes.

Así, mientras las computadoras de la primera generación, fueron instrumentos exclusivos de sus creadores y de algunos cuantos científicos, que requerían de meses de estudio y un alto grado de conocimientos previos de matemáticas, para construir programas relativamente simples, las máquinas de la segunda generación principiaron a estar al alcance de ingenieros y técnicos, gracias a la aparición de FORTRAN y de otros lenguajes similares, así como a la incorporación de los sistemas operativos, que habrían de relevar al programador de las tareas relacionadas con el control de las unidades periféricas.

Con la introducción de la terminal de consulta, del tiempo compartido y de nuevos lenguajes conceptualmente más simples, no sólo las tareas de programación se simplificaron, sino que

por primera vez, el personal no técnico (cajeros y empleados administrativos) pudieran entrar en contacto con la computadora, dejando atrás los tiempos en que las transacciones y los documentos eran enviados a los "centros de cálculo" o "unidades informáticas" para su proceso por medios misteriosos y fuera del control del usuario.

En la etapa de la cuarta generación, el proceso de acercamiento continuó, ahora las secretarias hacen uso de terminales para redactar oficios y cartas, los jefes tienden a depender cada vez más para comunicarse con sus empleados y consultar bancos de datos financieros, los empleados de hoteles e industrias hacen igualmente uso de ellas y en las escuelas, se convierten en un excelente apoyo para la experimentación y la enseñanza, el nivel de acercamiento es tal que aún los niños hacen uso cotidiano de ellas.

Factores de este fenómeno que habrá de intensificarse en el futuro, son la reducción de los costos, la confiabilidad y reducción de requerimientos para su operación y la concepción misma de los equipos de cómputo como intrumentos de uso masivo. El desarrollo de nuevos lenguajes interactivos y gráficos ha constituído un factor adicional de este proceso.

De este análisis es posible concluir que la interacción hombre/máquina habrá de continuar su crecimiento en las próximas generaciones de computadoras, buscando nuevas formas de acercamiento que hoy en día principian a delinearse.

b) La computación no numérica.

En sus orígenes, la computación fue totalmente numérica, no sólo en lo que respecta a la naturaleza de los datos y tipos de problemas que se pretendía resolver, sino al lenguaje mismo de programación.*

La dotación de caracteres alfabéticos a los mecanismos de impresión y el estudio de la linguística (Chomsky, Bar-Hillel, etc.) hizo factible el uso de lenguajes no numéricos para la programación, y la dependencia cada vez menor de los números, sin embargo, las aplicaciones fundamentales de la computación siguieron ligadas a los aspectos numéricos, durante las dos generaciones siguientes de computadoras.

Esta afirmación puede parecer extraña al lector, acostumbrado a pensar en aplicaciones administrativas en las que se hace referencia a nombres de personas, de empresas, de productos, o de lugares geográficos. Aunque esto es totalmente cierto,

* En las primeras computadoras, las instrucciones eran codificadas en forma totalmente numérica, de acuerdo con el modelo introducido por John Von Neumann, quién utilizó en su concepción de la computadora, las ideas expuestas por Kurt Godel que asociaba un número (programaba cada función computable).

es también válido mencionar que en todos estos casos la información alfabética nunca constituye la parte esencial del proceso; así por ejemplo, el nombre del empleado es sólo una etiqueta que se anexa al resultado del cálculo de percepciones y deducciones, en la generación de un documento de pago, y si bien es cierto que para la sociedad, quien importa es el portador del documento, para el computador el trabajo central reside en las cifras.

Esta observación no implica que el uso de la computadora fuese exclusivamente en aplicaciones numéricas, pues a partir de 1960, diversos esfuerzos de investigación se centraron en problemas no numéricos, logrando resultados que han tenido y tendrán una influencia notable en el desarrollo futuro de la computación.

Como ya se ha mencionado, el origen de las investigaciones sobre problemas no numéricos se situó en la lingüística formal, con la búsqueda de algoritmos que hicieran factible la traducción mecánica de lenguajes. Posteriormente, el desarrollo de LISP y otros lenguajes como Snobol y L^6, hicieron factibles los trabajos en inteligencia artificial y manipulación de símbolos algebráicos, estableciendo la posibilidad de dotar a las máquinas con capacidad de deducción para resolver problemas de lógica.

Es así que para los inicios de la década de los setenta existiesen máquinas capaces de jugar ajedrez, reconocer objetos a partir de su descripción linguística (para lo cual el computador requiere de la conexión de un equipo de televisión) y de hacer deducciones lógicas. Unos años más tarde las primeras máquinas capaces de sostener conversaciones sencillas, eran estudiadas en algunos institutos de investigación.

La construcción de máquinas con capacidad de inferencia y conversación generalizada, y la creación de "sistemas de conocimiento" como intrumento de uso masivo, constituye hoy en día el objetivo a lograr para la siguiente generación de computadoras.*

Otro proceso de desarrollo menos complejo, ha tenido lugar durante los últimos años en el campo de la computación no numérica y constituye hoy en día, la confirmación del éxito que habrán de tener los sistemas futuros de comunicación y conocimientos. Se trata de los populares "procesadores de palabra" que son utilizados en forma masiva, para la preparación de cartas y documentos técnicos.

* Figenbaun Conference on Fifth Generation Computer Systems. Tokio, 1981.

Con el procesador de palabras el usuario tiene acceso a un conjunto de instrucciones que le permiten editar textos, corregir errores y dar la presentación y organización más adecuada. El envío y almacenamiento de documentos con costos extraordinariamente reducidos, hace posible la utilización masiva de esta tecnología, que para los inicios de la década 80, ha principiado a substituir a la máquina de escribir y a los demás equipos tradicionales de oficina.

Sin lugar a dudas este proceso de orientación de la computación a las aplicaciones no numéricas representa una tendencia irreversible para el futuro.

c) La orientación de las investigaciones.

Como en otras áreas de la tecnología moderna, el período requerido para que los resultados exitosos de una investigación sean aplicados masivamente, se reduce, a unos cuantos meses o a lo más un par de años; es así que, el análisis de la evolución de los "objetos de estudio" relacionados con la computación, constituye una tercer área de análisis por su importancia.

A este respecto podemos decir que mientras en la primera generación, la preocupación y motivos de investigación, se orientaban al desarrollo de los métodos de cálculo numérico y a la búsqueda de nuevos dispositivos de almacenamiento magnético,

durante la siguiente etapa los temas de estudio estuvieron centrados en la linguística, la utilización de transistores y circuitos integrados, y en la naturaleza de los procesos administrativos.

Al iniciarse la era de la tercera generación de computadoras, el interés de la investigación varió hacia el análisis del funcionamiento de las organizaciones (con el enfoque de sistemas), al uso de la tecnología LSI, al desarrollo de los sistemas de información y hacia la inteligencia artificial.

Finalmente, los temas de interés al entrar al mercado la cuarta generación, eran el estudio de la sociedad y las interacciones que en su seno tenían los organismos industriales y financieros y por otro lado las nuevas formas de aplicación de la micro-electrónica, y el estudio de las comunicaciones digitales.

Actualmente y para el futuro, el tema principal de estudio ha cambiado nuevamente para centrarse en "el hombre", en los procesos cognitivos, de la inferencia y la acumulación de conocimientos. Los avances de la micro-electrónica hacen factible imitar los procesos adaptivos de los organismos biológicos y la posibilidad de construir mosaicos de pequeños autómatas para formar "tejidos" cuya capacidad de asimilación y análisis de imágenes, sea similar a la del ojo humano, reconocido ya como uno de los sistemas más eficientes de recolección de información.

5. LAS COMPUTADORAS DE LA QUINTA GENERACION.

Con estas nuevas orientaciones y con un mercado de dimensiones jamás superadas en su magnitud y diversificación, se principian a definir en los centros de investigación norteamericanos y europeos, los conceptos que darán lugar a la quinta generación de computadoras.

Paralelamente y quizás con un empuje mayor, la industria japonesa desarrolla un proyecto de gran magnitud para poner en operación una nueva generación de computadoras substancialmente distintas a las existentes, en los primeros años de la década de 1990.*

Estas nuevas máquinas habrán de caracterizarse por la utilización de enjambres procesadores microscópicos, operando simultáneamente para recibir y clasificar información, por su capacidad básica de inferencia y generación de "conocimientos" y esquemas generales, a partir de información particular, así como por su estrecha relación con el hombre.

* Este proyecto fue presentado por primera ocasión en la Conferencia sobre "Computadora de la Quinta Generación", realizada en octubre de 1981, mencionada previamente en este reporte.

El sistema de control habrá de seguir y utilizar principios ya conocidos hoy en día, sólo se usan en experimentos de inteligencia artificial, como los relacionados con las máquinas LISP y de Flujo de datos (Data flow machines).

El desarrollo de la computación no numérica como forma principal de aplicación, y la interacción más intensa entre hombre y máquina estarán también en el centro del proceso. Finalmente, la importancia de la comunicación entre máquinas, y con ello, entre los hombres que la poseen, habrán de delinear en buena medida el futuro de la civilización.

6. NUESTRA PARTICIPACION EN EL PROCESO.

Cuál es y será nuestro papel en esta revolución ¿la de observadores pasivos del proceso? ¿Pasará inadvertido como otros tantos cambios tecnológicos de cuyas implicaciones políticas y sociales nos enteramos sólo años después? ¿Podríamos ser capaces de asimilar esta vez las tecnologías emergentes? ¿Qué planes podemos formular para tomar una parte activa por modesta que sea en este proceso de cambio? Quizás estas sean algunas preguntas que hoy deberíamos hacernos.

DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA U.N.A.M.

LENGUAJE DE PROGRAMACION BASIC - CON APLICACIONES

(PRIMERA PARTE)

SISTEMAS DE NUMERACION

FEBRERO, 1983

Palacio de Minería Calle de Tacuba 5 primer piso Deleg. Cuauhtemoc 06000 México, D.F. Tel.: 521-40-20 Apdo. Postal M-2285

S I S T E M A S D E N U M E R A C I O N

Ing. Heriberto Olguín Romo

Un número cualquiera N, puede ser expresado en la forma:

$$N = a_0 r^0 + a_1 r^1 + a_2 r^2 + \ldots + a_n r^n, \quad -\infty < n < \infty$$

donde:

r es la base del sistema de numeración.

$a_0, a_1, a_2, \ldots, a_n$ son números dígitos que pueden tener valores de 0 a (r-1).

Así, cuando escribimos un número decimal, digamos 1967, lo estamos escribiendo en una forma abreviada, puesto que:

$$1967 = 7 \times 10^0 + 6 \times 10^1 + 9 \times 10^2 + 1 \times 10^3$$

Notamos que en el sistema decimal, los dígitos usados son los que se encuentran entre el 0 y el 9. Si usamos una base r < 10, tendremos que usar solamente los dígitos familiares de 0 a (r-1). Si r > 10, tendremos que idear dígitos nuevos para representar 10, 11, 12, ..., (r-1).

Los métodos para cambiar un número expresado en términos de otra base cualquiera se discuten en numerosos libros de álgebra ordinaria. Para nuestro caso discutiremos únicamente los sistemas:

Binario y Octal y las relaciones existentes entre éstos y las existentes con el sistema decimal (ya conocido) y viceversa.

a) . Sistema Binario.

Un número escrito en base 2 toma la forma:

$$N = a_0 2^0 + a_1 2^1 + a_2 2^2 + a_3 2^3 + \ldots + a_n 2^n$$
$$= a_0 + a_1(2) + a_2(4) + a_3(8) + \ldots + a_n(2^n)$$

Los coeficientes $a_0, a_1, a_2, a_3, \ldots, a_n$ pueden tener únicamente los valores 0 o 1, puesto que, (r-1) = 2-1 = 1.

Si dividimos N_2 por la base 2, cada potencia de 2 se reduce en uno y queda un término que será el residuo, a_0. Si $N_2/2$, con el residuo a_0 descartado, se divide nuevamente por 2, obtenemos un nuevo residuo a_1. Continuando con este proceso de descarte de residuos previo y dividiendo por 2, se obtiene una sucesión de residuos que resultan ser los coeficientes a_0, a_1, a_2, a_3, ..., a_n; y arreglando estos coeficientes en el orden: a_n, ..., a_3, a_2, a_1, a_0; se obtiene la forma convencional del número binario deseado; con los dígitos de orden superior a la izquierda y en forma sucesiva a la derecha va decreciendo el orden de dichos dígitos.

Podemos, mediante lo explicado anteriormente, cambiar el número 175 en base decimal, a un número en base binaria mediante divisiones sucesivas, de la siguiente manera; en la siguiente página se muestra este ejemplo:

DIVISIONES SUCESIVAS

COCIENTE	RESIDUO
175	
87	1
43	1
21	1
10	1
5	0
2	1
1	0
0	1

Si ahora, la columna de residuos es leída de abajo hacia arriba y arreglada en orden convencional, podemos entonces escribir el binario equivalente al decimal 175:

$$175 = 10101111$$

Otro método, que algunas veces es más rápido, es mediante la sustracción de potencias de 2, del número decimal, comenzando siempre con el número que represente a 2 elevado al exponente de mayor valor que pueda sustraerse del número decimal, y del resultado de esa sustracción, sustraer nuevamente otro número, decreciente en el orden del exponente de 2; y así sucesivamente hasta llegar a $2^0 = 1$. Cada vez que pueda ser extraída una potencia de 2, un dígito 1 existirá en el equivalente binario.

Si una potencia de 2 no puede ser sustraída, existirá el dígito 0. Llevemos a cabo mediante este procedimiento el ejemplo anterior:

175		
- 128	$128 = 2^7$	puede sustraerse
47		
- 32	$32 = 2^5$	" "
15		
- 8	$8 = 2^3$	" "
7		
- 4	$4 = 2^2$	" "
3		
- 2	$2 = 2^1$	" "
1		
- 1	$1 = 2^0$	" "
0		

En este ejemplo notamos que 2^5 y 2^4 no pueden sustraerse, así que el binario equivalente del decimal 175 será:

$$175 = 10101111$$

La transformación de un número binario a su equivalente en decimal es realizada simplemente sumando las potencias de 2 correspondientes a la posición del dígito, que contiene unos (1s) en la representación binaria.

De tal manera que el decimal equivalente de 10101111 será:

$$128 + 32 + 8 + 4 + 2 + 1 = 175$$

Operaciones Binarias.

Las reglas de la adición, sustracción, multiplicación y división, se ilustran a continuación mediante tablas correspondientes a cada una de las operaciones antes citadas, y ejemplos de cada una de éstas, acompañados de ejemplos correspondientes en el sistema decimal, nótese que se comprueba la operación binaria con la decimal.

Adición

+	0	1
0	0	1
1	1	0

→ y un acarreo

Sustracción

-	0	1
0	0	1
1	0	0

↓ y una deuda

1100001	97
+ 1001110	+ 78
10101111	175

10101111	175
- 1001011	- 75
01100100	100

Multiplicación

x	0	1
0	0	0
1	0	1

División

÷	0	1
0	?	?
1	0	1

11001	25
x 111	x 7
11001	175
11001	
11001	
10101111	

	100011		35
101	10101111	5	175
	000		25
	0111		0
	0101		
	000		

Cambio de un decimal fraccionario a binario:

0.4375 x 2 = 0.8750	0.0	
0.8750 x 2 = 1.7500	0.01	$0.4375 = 0.0111_2$
0.75 x 2 = 1.5	0.011	
0.5 x 2 = 1.0	0.0111	

Cambio de un número binario fraccionario a decimal:

$$0.0111 = 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4}$$

$$= \frac{1}{4} + \frac{1}{8} + \frac{1}{16} = \frac{4 + 2 + 1}{16} = \frac{7}{16} = 0.4375$$

b) Sistema Octal.

Un número escrito en base 8 toma la forma:

$$N = a_0 8^0 + a_1 8^1 + a_2 8^2 + \ldots + a_n 8^n$$
$$= a_0 + a_1(8) + a_2(16) + \ldots + a_n(8^n)$$

Los coeficientes $a_0, a_1, a_2, \ldots a_n$ pueden tener los valores: 0, 1, 2, 3, 4, 5, 6 y 7, ya que (r-1) = 8 - 1 = 7.

Procediendo en la misma forma como en el sistema binario podemos cambiar del sistema decimal al octal; ilustraremos mediante un ejemplo, el cambio del decimal 1967 a octal:

```
      245
8 | 1967      7
    36
     47       .
      7

      30
8 | 245       5
    05

       3
8 |  30       6
     6

       0
8 |   3       3
      3
```

1 9 6 7 = 3 6 5 7 8

La transformación de un número octal a su equivalente en decimal, es realizada sumando las potencias de 8 correspondientes a la posición del dígito y multiplicar cada una por el dígito en cuestión, de tal manera que el decimal equivalente del octal 3657 será:

$$3 \times 8^3 + 6 \times 8^2 + 5 \times 8^1 + 7 \times 8^0 = 3 \times 512 + 6 \times 64 + 40 + 7$$
$$= 1536 + 384 + 47 = 1967$$

$$3657_8 = 1967$$

Para cambiar de sistema binario a octal, procedemos haciendo grupos de tres dígitos binario de derecha a izquierda, y poniendo cada uno de esos grupos en su equivalente en octal mediante la siguiente tabla:

DECIMAL	BINARIO	OCTAL
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	10
9	1001	11
10	1010	12

Sea cambiar el binario (1010111) a octal:

1 0 1 0 1 1 1

2 5 7

$1010111_2 = 257_8$

Para cambiar de octal a binario procedemos en forma contraria a la anterior, o sea: cambiar el octal 175 a binario:

1 7 5

1 1 1 1 0 1

$175_8 = 1111101_2$

Operaciones Octales.

Las reglas de la adición y multiplicación se dan en las siguientes tablas:

Adición

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

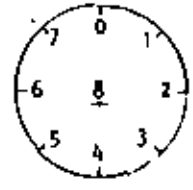
Multiplicación

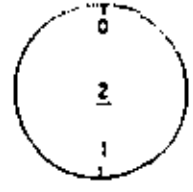
X	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	10	12	14	16
3	0	3	6	11	14	17	22	25
4	0	4	10	14	20	24	30	34
5	0	5	12	17	24	31	36	43
6	0	6	14	22	30	36	44	52
7	0	7	16	25	34	43	52	61

Octal	Decimal	Octal	Decimal
3657	1967	175	125
+ 273	+ 187	X 32	X 26
4152	2154	372	750
		567	250
		6262	3250

Cualquier sistema de numeración lo podemos representar en forma cíclica. Un círculo se divide en tantas partes como lo especifique el sistema de numeración empleado; hay que tomar en cuenta que cada ciclo completo equivale a un acarreo.

Los ciclos para los sistemas Decimal, Octal y Binario se ilustran en las siguientes figuras:





En las computadoras digitales los sistemas de numeración más empleados son:

a) Sistema Binario.

b) Sistema Octal.

c) Sistema Decimal.

d) Sistema Hexadecimal.

Hemos visto que el sistema binario consta de dos caracteres (0,1), el octal de ocho caracteres (0, 1, 2, 3, 4, 5, 6 y 7), el decimal de diez caracteres (0, 1, 2, 3, 4, 5, 6, 7, 8 y 9); el sistema hexadecimal constará entonces de 16 caracteres que son: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E y F.

Casos concretos de computadoras es la del Centro de Cálculo de la Facultad de Ingeniería de la U. N. A. M. que tiene una Digital VAX-11/780 que usa el sistema binario.

S I S T E M A S D E N U M E R A C I O N

Ing. Heriberto Olguín Romo

Un número cualquiera N, puede ser expresado en la forma:

$$N = a_0 r^0 + a_1 r^1 + a_2 r^2 + \ldots + a_n r^n, \quad -\infty < n < \infty$$

donde:

r es la base del sistema de numeración.

$a_0, a_1, a_2, \ldots, a_n$ son números dígitos que pueden tener valores de 0 a (r-1).

Así, cuando escribimos un número decimal, digamos 1967, lo estamos escribiendo en una forma abreviada, puesto que:

$$1967 = 7 \times 10^0 + 6 \times 10^1 + 9 \times 10^2 + 1 \times 10^3$$

Notamos que en el sistema decimal, los dígitos usados son los que se encuentran entre el 0 y el 9. Si usamos una base $r < 10$, tendremos que usar solamente los dígitos familiares de 0 a (r-1). Si $r > 10$, tendremos que idear dígitos nuevos para representar 10, 11, 12, ..., (r-1).

Los métodos para cambiar un número expresado en términos de otra base cualquiera se discuten en numerosos libros de álgebra ordinaria. Para nuestro caso discutiremos únicamente los sistemas:

Binario y Octal y las relaciones existentes entre éstos y las existentes con el sistema decimal (ya conocido) y viceversa.

a) Sistema Binario.

Un número escrito en base 2 toma la forma:

$$N = a_0 2^0 + a_1 2^1 + a_2 2^2 + a_3 2^3 + \ldots + a_n 2^n$$
$$= a_0 + a_1(2) + a_2(4) + a_3(8) + \ldots + a_n(2^n)$$

Los coeficientes a_0, a_1, a_2, a_3, ..., a_n pueden tener únicamente los valores 0 o 1, puesto que, $(r-1) = 2-1 = 1$.

Si dividimos N_2 por la base 2, cada potencia de 2 se reduce en uno y queda un término que será el residuo, a_0. Si $N_2/2$, con el residuo a_0 descartado, se divide nuevamente por 2, obtenemos un nuevo residuo a_1. Continuando con este proceso de descarte de residuos previo y dividiendo por 2, se obtiene una sucesión de residuos que resultan ser los coeficientes a_0, a_1, a_2, a_3, ..., a_n; y arreglando estos coeficientes en el orden: a_n, ..., a_3, a_2, a_1, a_0; se obtiene la forma convencional del número binario deseado; con los dígitos de orden superior a la izquierda y en forma sucesiva a la derecha va decreciendo el orden de dichos dígitos.

Podemos, mediante lo explicado anteriormente, cambiar el número 175 en base decimal, a un número en base binaria mediante divisiones sucesivas, de la siguiente manera; en la siguiente página se muestra este ejemplo:

D I V I S I O N E S S U C E S I V A S

COCIENTE	RESIDUO
175	
87	1
43	1
21	1
10	1
5	0
2	1
1	0
0	1

Si ahora, la columna de residuos es leída de abajo hacia arriba y arreglada en orden convencional, podemos entonces escribir el binario equivalente al decimal 175:

1 7 5 = 1 0 1 0 1 1 1 1

Otro método, que algunas veces es más rápido, es mediante la sustracción de potencias de 2, del número decimal, comenzando siempre con el número que represente a 2 elevado al exponente de mayor valor que pueda sustraerse del número decimal, y del resultado de esa sustracción, sustraer nuevamente otro número, decreciente en el orden del exponente de 2; y así sucesivamente hasta llegar a $2^0 = 1$. Cada vez que pueda ser extraída una potencia de 2, un dígito 1 existirá en el equivalente binario.

Si una potencia de 2 no puede ser sustraída, existirá el dígito 0. Llevemos a cabo mediante este procedimiento el ejemplo anterior:

175		
- 128	$128 = 2^7$	puede sustraerse
47		
- 32	$32 = 2^5$	" "
15		
- 8	$8 = 2^3$	" "
7		
- 4	$4 = 2^2$	" "
3		
- 2	$2 = 2^1$	" "
1		
- 1	$1 = 2^0$	" "
0		

En este ejemplo notamos que 2^6 y 2^4 no pueden sustraerse, así que el binario equivalente del decimal 175 será:

$$1\ 7\ 5 = 1\ 0\ 1\ 0\ 1\ 1\ 1\ 1$$

La transformación de un número binario a su equivalente en decimal es realizada simplemente sumando las potencias de 2 correspondientes a la posición del dígito, que contiene unos (1s) en la representación binaria.

De tal manera que el decimal equivalente de 10101111 será:

$$128 + 32 + 8 + 4 + 2 + 1 = 175$$

Operaciones Binarias.

Las reglas de la adición, sustracción, multiplicación y división, se ilustran a continuación mediante tablas correspondientes a cada una de las operaciones antes citadas, y ejemplos de cada una de éstas, acompañados de ejemplos correspondientes en el sistema decimal, nótese que se comprueba la operación binaria con la decimal.

Adición

+	0	1
0	0	1
1	1	0 → y un acarreo

Sustracción

-	0	1
0	0	1
1	0	0

↓
y una deuda

1100001	97
+ 1001110	+ 78
10101111	175

10101111	175
- 1001011	- 75
01100100	100

Multiplicación

X	0	1
0	0	0
1	0	1

División

÷	0	1
0	?	?
1	0	1

```
   11001     25
X    111   X  7
-------   -----
   11001    175
  11001
 11001
-------
10101111
```

```
          100011      35
101 | 10101111   5 | 175
     000              25
       0111            0
        0101
         000
```

Cambio de un decimal fraccionario a binario:

0.4375 X 2 = 0.8750	0.0	
0.8750 X 2 = 1.7500	0.01	$0.4375 = 0.0111_2$
0.75 X 2 = 1.5	0.011	
0.5 X 2 = 1.0	0.0111	

Cambio de un número binario fraccionario a decimal:

$$0.0111 = 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4}$$

$$= \frac{1}{4} + \frac{1}{8} + \frac{1}{16} = \frac{4 + 2 + 1}{16} = \frac{7}{16} = 0.4375$$

b) Sistema Octal.

Un número escrito en base 8 toma la forma:

$$N = a_0 8^0 + a_1 8^1 + a_2 8^2 + \ldots + a_n 8^n$$
$$= a_0 + a_1(8) + a_2(16) + \ldots + a_n(8^n)$$

Los coeficientes a_0, a_1, a_2, ... a_n pueden tener los valores: 0, 1, 2, 3, 4, 5, 6 y 7, ya que (r-1) = 8 - 1 = 7.

Procediendo en la misma forma como en el sistema binario podemos cambiar del sistema decimal al octal; ilustraremos mediante un ejemplo, el cambio del decimal 1967 a octal:

```
   245
8 | 1967      7
    36
     47
      7

    30
8 | 245       5
    05                    1967 = 3657₈

     3
8 | 30        6
     6

     0
8 | 3         3
     3
```

La transformación de un número octal a su equivalente en decimal, es realizada sumando las potencias de 8 correspondientes a la posición del dígito y multiplicar cada una por el dígito en cuestión, de tal manera que el decimal equivalente del octal 3657 será:

$$3 \times 8^3 + 6 \times 8^2 + 5 \times 8^1 + 7 \times 8^0 = 3 \times 512 + 6 \times 64 + 40 + 7$$
$$= 1536 + 384 + 47 = 1967$$

$$3657_8 = 1967$$

Para cambiar de sistema binario a octal, procedemos haciendo grupos de tres dígitos binario de derecha a izquierda, y poniendo cada uno de esos grupos en su equivalente en octal mediante la siguiente tabla:

DECIMAL	BINARIO	OCTAL
0	0000	0
1	0001	1
2	0010	2
3	0011	3
4	0100	4
5	0101	5
6	0110	6
7	0111	7
8	1000	10
9	1001	11
10	1010	12

Sea cambiar el binario 10101111 a octal:

1 0 1 0 1 1 1 1

2 5 7

$10101111_2 = 257_8$

Para cambiar de octal a binario procedemos en forma contraria a la anterior, o sea: cambiar el octal 175 a binario:

1 7 5

1 1 1 1 1 0 1

$175_8 = 1111101_2$

Operaciones Octales.

Las reglas de la adición y multiplicación se dan en las siguientes tablas:

Adición

+	0	1	2	3	4	5	6	7
0	0	1	2	3	4	5	6	7
1	1	2	3	4	5	6	7	10
2	2	3	4	5	6	7	10	11
3	3	4	5	6	7	10	11	12
4	4	5	6	7	10	11	12	13
5	5	6	7	10	11	12	13	14
6	6	7	10	11	12	13	14	15
7	7	10	11	12	13	14	15	16

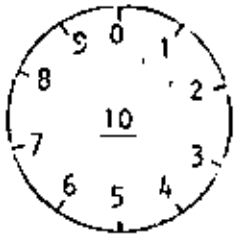
Multiplicación

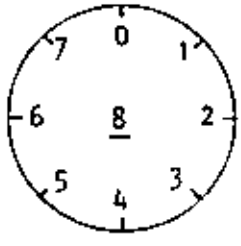
X	0	1	2	3	4	5	6	7
0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7
2	0	2	4	6	10	12	14	16
3	0	3	6	11	14	17	22	25
4	0	4	10	14	20	24	30	34
5	0	5	12	17	24	31	36	43
6	0	6	14	22	30	36	44	52
7	0	7	16	25	34	43	52	61

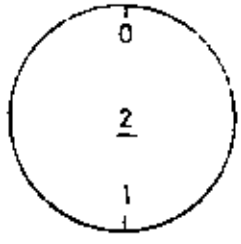
Octal	Decimal	Octal	Decimal
3657	1967	175	125
+ 273	+ 187	X 32	X 26
4152	2154	372	750
		567	250
		6262	3250

Cualquier sistema de numeración lo podemos representar en forma cíclica. Un círculo se divide en tantas partes como lo especifique el sistema de numeración empleado; hay que tomar en cuenta que cada ciclo completo equivale a un acarreo.

Los ciclos para los sistemas Decimal, Octal y Binario se ilustran en las siguientes figuras:







En las computadoras digitales los sistemas de numeración más empleados son:

a) Sistema Binario.

b) Sistema Octal.

c) Sistema Decimal.

d) Sistema Hexadecimal.

Hemos visto que el sistema binario consta de dos caracteres (0,1), el octal de ocho caracteres (0, 1, 2, 3, 4, 5, 6 y 7), el decimal de diez caracteres (0, 1, 2, 3, 4, 5, 6, 7, 8 y 9); el sistema hexadecimal constará entonces de 16 caracteres que son: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E y F.

Casos concretos de computadoras es la del Centro de Cálculo de la Facultad de Ingeniería de la U. N. A. M. que tiene una Digital VAX-11/780 que usa el sistema binario.

DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA U.N.A.M.

LENGUAJE DE PROGRAMACION BASIC - CON APLICACIONES
(PRIMERA PARTE)

EL UNIVERSO DE LAS COMPUTADORAS

FEBRERO, 1983

Palacio de Minería Calle de Tacuba 5 primer piso Deleg. Cuauhtemoc 06000 México, D.F. Tel.: 521-40-20 Apdo. Postal M-2285

EL UNIVERSO DE LAS COMPUTADORAS

Por: Lic. Marcia de las Fuentes

(Tomado de la revista Geografía Universal, Año 7, Vol. 13, No. 6)

Mientras el público observaba con curiosidad, Bouchon colocó la hoja de papel con pequeños agujeros en el rodillo del telar. El cilindro comenzó a girar y la gente lanzó una exclamación de asombro: como por arte de magia, de la máquina fue surgiendo un hermoso tejido perfectamente diseñado en seda. La demostración había tenido buen éxito.

Basile Bouchon fue quien construyó en 1725 el primer telar que podía tejer siguiendo la clave cifrada en una hoja de papel perforado. El método era sencillo y práctico; consistía en realizar agujeros en un rollo de papel siguiendo el diseño que se deseaba tejer. Cuando este papel se presionaba contra una hilera de agujas, las que coincidían con los agujeros permanecían en la misma posición; las otras se movían hacia adelante. De tal forma iba lográndose el dibujo en el tejido. De esta manera nacía el primer "diálogo" entre la máquina y el hombre; comunicación que habría de convertirse, siglos más tarde, en un fundamento de la ciencia y permitiría enormes avances para la tecnología.

La historia de los cerebros electrónicos es muy reciente, aunque sus primitivos antecesores hayan sido creados hace muchos siglos.

Es probable que fuese en Babilonia, 5,000 años atrás, cuando el proceso de contar con los dedos sufrió su primera modificación. Seguramente algún anónimo comerciante, confundido con su dinero, creó el ábaco, ese rudimentario pero efectivo sistema de contabilización. Transmitido a través de todas las civilizaciones, el ábaco se convirtió en el tradicional instrumento que introduce a los niños en el complejo mundo de los números. Los chinos desarrollaron y refinaron ese calculador, y a tal punto lo hicieron, que aún existen algunos profesionales capaces de realizar cuentas con más velocidad que las máquinas de sumar mecánicas.

Pero el ábaco, aunque mucho más efectivo que el viejo sistema de contar con los dedos, está basado en la memoria visual del individuo que lo acciona. Este debe recordar cuántos son los pequeños discos que ha movido -y si lo olvida, volverlos a contar-, para sumarlos o restarlos con las otras unidades ubicadas en la siguiente barra. Necesita, por lo tanto, concentrarse en la operación y dedicar su atención a ella.

En 1612, el filósofo Blaise Pascal inventó una máquina de sumar y restar que era muy superior a la precedente. Se trataba de una pequeña caja que encerraba en su interior cilindros y engranajes; las ruedas de la parte superior del aparato correspondían a las unidades, decenas, centenas y subsiguientes, y cada rueda registraba de cero a nueve. El invento fue valioso aunque poco práctico por sus características.

Más eficientes resultaron los Rodillos de Napier, fabricados por John Napier y que servían para multiplicar. La nobleza europea del siglo XVII los recibió con mucho entusiasmo debido al pequeño tamaño que facilitaba el traslado. Eran rodillos que contenían los dígitos del 1 al 9, con sus múltiples en columnas debajo de ellos. Al hacer girar los rodillos se podía multiplicar fácilmente y sin demorar mucho tiempo. Otro aporte fundamental lo realizó el inglés Charles Babbage, quien entre 1812 y 1822 ideó y realizó un complicado artefacto que podía calcular y hasta imprimir las tablas matemáticas.

UN NUEVO LENGUAJE

La automación, palabra derivada de automatización y acuñada por el norteamericano Delmar S. Harder, está dirigida a reducir el esfuerzo y dejar que el control mecánico o electrónico sustituya el control que ejerce el cerebro del hombre. Existen, por supuesto, diversos niveles de mecanización; nadie se asombra cuando una aspiradora recoge en pocos segundos el polvo acumulado en el piso y evita el ejercicio manual del barrido. El hábito en el uso de los numerosos artefactos que la industria provee al hogar, ha generado cierta indiferencia en cuanto a su mecanismo. Según expertos, hacia finales de la actual década, los países desarrollados contarán con casi 80 máquinas de distinto tipo para cada uno de los hogares. Estos aparatos, que podrían ubicarse en una categoría inferior, están dedicados a cumplir diversas tareas domésticas.

Pero existe un nivel superior en la automación; ese nivel se llama electrónica y gracias a ella el desarrollo de las disciplinas humanas ha alcanzado una eficiencia admirable.

Este alto nivel de las máquinas se encuentra en las computadoras digitales, perfectos cerebros que no sólo son capaces de controlar su propio funcionamiento, sino también de dirigir y comunicarse con otras máquinas; recibir la información que le envían y procesarla, alertar sobre las posibles deficiencias y subsanarlas en caso de que se produzcan.

No obstante que tales aparatos son de uso corriente en prácticamente todas las disciplinas científicas y técnicas, la mayoría de las personas no ha asimilado aún el verdadero carácter que poseen y observan con incredulidad y hasta cierta desconfianza a estos robots infalibles, presuntos competidores de los hombres.

Una cosa es apretar un botón y que la licuadora, la lavadora o la máquina de afeitar se pongan en funcionamiento, y otra muy distinta es que exista una comunicación directa con un aparato capaz de responder a los interrogantes planteados por un ser humano. En el segundo caso se establece un diálogo entre el hombre y la máquina, y es precisamente eso lo que provoca ciertas reservas en aquellos que temen ver al mundo dominado por cerebros electrónicos.

Pero saliendo de este terreno sin asidero real, el hecho de que se haya encontrado un método de comunicación entre el hombre y el objeto posibilitó que la ciencia avance a pasos gigantescos.

Para lograr esa relación fue necesario crear un nuevo lenguaje, que pudiera ser comprendido por las computadoras y que a su vez le permitiera al hombre recibir los mensajes que ella envía. Ese nuevo idioma se llama cibernética y es una disciplina dedicada a la comunicación entre hombre y máquina, máquina y hombre y máquina y máquina. La palabra deriva del griego kybernetes, y define los mensajes intercambiables que forman la comunicación recíproca.

El problema residía en encontrar un vocabulario apropiado para la computadora, que le permitiera responder a las preguntas. Bouchon, con su telar de papeles perforados, resultó el promotor de ese lenguaje. Los agujeros o no agujeros que iban encontrando las agujas a su paso fueron los antecedentes de las tarjetas perforadas que hoy se utilizan.

En realidad, se trata de un vocabulario simple basado en el sí o no del sistema binario o de dos bases. El lenguaje de un foco de luz, por ejemplo, consiste en encendido y apagado; de igual forma, las tarjetas que se introducen en la computadora son leídas por ellas mismas casi como en el telar de Bouchon: la corriente eléctrica penetra en el agujero de la ficha o no lo hace.

Puesto que la computadora se limita a la respuesta de si ha penetrado o no, los números que se introducen deberán ser perforados en las fichas mediante el código binario expresado en agujeros y espacios.

Ese es el sistema básico y a partir de él se crearon otros métodos que siguen el mismo procedimiento; uno de ellos es la cinta perforada y el otro la cinta magnetofónica. En este último caso, cada cinta tiene siete canales en los que se encuentran puntos magnetizados con cabezas de electroimanes. Al pasar esos puntos por el mecanismo lector, se convierten en pulsaciones eléctricas que van traduciendo el mensaje. Esto se realiza a una velocidad de 630,000 puntos por segundo.

Las computadoras no sólo han reducido el esfuerzo humano en la industria y la investigación, sino que además han posibilitado una mayor rapidez en los procesos algebraicos y han eliminado el margen de error. La mayoría de cerebros electrónicos está dotada del sistema "feedback". Estas máquinas poseen un autocontrol capaz de corregir sus propias deficiencias y las de aquellas que estén bajo su dirección. En caso de que una comience a funcionar mal, el cerebro principal recibirá de inmediato la señal de alarma y buscará en su memoria cuales son los procedimientos adecuados para corregir el problema. Si está en condiciones de solucionarlo, lo hará sin que tenga que intervenir ningún hombre. En caso contrario detendrá el funcionamiento de la máquina descompuesta y avisará al operador qué sucede y dónde debe dirigirse para encontrar la falla.

EL "PENSAMIENTO" ELECTRONICO

La pregunta surge espontáneamente: ¿cómo un aparato compuesto por circuitos eléctricos y sistemas magnéticos puede "pensar" o tener la suficiente autonomía a fin de controlar el funcionamiento de otras máquinas sin la presencia humana?

En primer término, debe recordarse que las computadoras no pueden realizar nada que no haya sido programado previamente por el hombre. Son cerebros electrónicos que sólo se ponen en funcionamiento cuando se les suministran los elementos necesarios para que lo hagan. Y no hay nada que los asemeje a una inteligencia artificial.

Para que esas máquinas cumplan con su tarea, es necesario alimentarlas; se les da el problema y la información que necesitarán a fin de solucionarlo. A partir de ese momento, la información suministrada pasará al sistema de control y al sistema de memoria.

El primero toma la información y la organiza para su posterior selección. El segundo comprueba que todos los datos estén correctos y que no haya error alguno; en caso de que advierta una equivocación, avisará cuál es y dónde está. Por ejemplo, si se ha introducido en la computadora un texto y hay una frase en la cual se abre un signo de paréntesis que luego no cierra, el control dará la alarma. Es que el sistema observó la apertura del signo y mientras continúa recorriendo las palabras siguientes espera la llegada del cierre, porque ha sido programado para que cada vez que se abra un paréntesis se cierre posteriormente; si así no ocurre es porque hay un

El sistema de memoria es el que recibe toda la información y la almacena en sus unidades. Le servirá de antecedente cada vez que tenga que retornar al mismo caso. Allí se mantienen todos los datos clasificados y listos para ser utilizados cuando el control los necesite. Existe también la biblioteca, que es el lugar donde se guardan los métodos para solucionar problemas. Por medio de circuitos, este sistema brinda las instrucciones básicas que previamente le han dado los operadores.

El paso siguiente es la solución del problema. En forma distinta que el cerebro humano, la computadora actúa por repetición, con la lógica suministrada por un programador humano.

Finalmente, la máquina otorga la respuesta de acuerdo con el sistema en que opere: fichas perforadas, cintas magnéticas, hojas escritas a máquina.

La explicación más elemental de una computadora podrá realizarse de la siguiente forma: a) suministro de información (alimentación); b) almacenamiento de esa información (memoria); c) solución al problema (elaboración); d) respuesta final del resultado del problema.

DIVERSAS FUNCIONES

Las necesidades científicas han impuesto una amplia diversificación entre las mismas computadoras, de acuerdo con las funciones que deben cumplir. Existen dos clases de máquinas; la denominada Analog (vocablo derivado del griego análogos) y la Digital, del latín digitus o "dedos", así llamado por la costumbre de contar con los dedos.

Las primeras no se ocupan de los números sino de cantidades físicas análogas; su trabajo consiste en expresarse en términos físicos y no numéricos, como por ejemplo, el ángulo de rotación de un eje, el voltaje eléctrico, etcétera. En cambio, las digitales se dedican a calcular y computar numéricamente; viven de pulsaciones eléctricas que recuerdan al antiguo sistema de Bouchon y son capaces de resolver complicados problemas algebraicos con mucha mayor velocidad que 500,000 hombres trabajando simultáneamente, no con papel y lápiz, sino con calculadoras manuales.

Los ingenieros que programaron los viajes espaciales y el primer descenso del hombre en la Luna, admitieron que todas esas actividades hubieran sido imposibles de realizar si no contaban con las modernas computadoras. Ningún ser humano está en condiciones de calcular las trayectorias, la propulsión y las necesidades de combustible con la suficiente exactitud como para hacer posible el experimento.

Los millones de sumas y restas requeridos para calcular los constantes cambios de gravitación de la Tierra, la Luna y el Sol, hubieran ocupado la actividad de varios miles de ingenieros que, de todos modos, habrían tardado varios siglos en llegar a conclusiones no del todo exactas.

Si bien lo anterior da una idea aproximada de la labor que cumplen estas criaturas electrónicas, no menos significativo es recordar que en la mayoría de los países las tareas de mantenimiento de servicios están a cargo de computadoras. Son ellas quienes controlan el suministro de electricidad y avisan sobre las reparaciones que deberán hacerse en el futuro, las que regulan el uso de agua potable en las ciudades o informan a los fabricantes de automóviles acerca de las tendencias del mercado y de las necesidades que se deberán tomar en cuenta. Aconsejan a los productores de manzanas de Nueva Inglaterra sobre los períodos de cosecha más óptimos, recopilan fórmulas de mezclas para fabricantes de piensos de ganado vacuno y de aves o ayudan a los médicos a determinar las dosis de radiación para los enfermos de cáncer.

La meteorología, considerada siempre como un arte de la predicción con un alto margen de error, ha logrado progresos notables gracias al funcionamiento de estos cerebros. Antes de constituirse en ciencia, estaba basada en las impresiones personales de los campesinos, el comportamiento de los pájaros o el dolor reumático de alguna anciana. A través de ellos se efectuaban los pronósticos del tiempo y sus posibles variaciones; la lluvia, la humedad, la sequía eran "advertidas" mediante signos que brindaba la propia naturaleza e "interpretados" arbitrariamente por los hombres.

El meteorólogo, científicamente, medía la presión atmosférica, observaba las nubes y hacía algunos pequeños experimentos que le proporcionaban datos, para determinar los cambios futuros de clima. Hoy existen computadoras electrónicas que pueden efectuar un millón de cálculos por segundo y que son capaces de anunciar las lluvias o las sequías conforme al análisis de información suministrada.

El Centro Meteorológico Nacional -NMC- de los Estados Unidos, recibe cuatro veces por día los datos enviados por varios satélites en órbita, así como por 2,000 estaciones meteorológicas de todo el mundo, más 3,200 informes de aviones comerciales y alrededor de 200 reportes elaborados (por computadoras) en vuelos de reconocimiento. Sería imposible que ese caudal de datos fuera recopilado, estudiado e interpretado por los hombres; en cambio, un enorme complejo de computadoras se encarga de hacerlo. Devora miles de informes sobre el tiempo, los selecciona y realiza millones de cálculos en un lapso de 90 minutos; luego saca conclusiones y las entrega. Actualmente, las predicciones para las 18 horas siguientes se consideran exactas en un 85 por ciento y se calcula que con las pequeñas computadoras que ya están circulando en los satélites alrededor de la Tierra y suministrando más información, en los próximos años se obtendrán conocimientos suficientes como para hacer predicciones absolutamente ciertas.

En este caso, los cerebros electrónicos no se limitan a suministrar los datos sino también "graficarlos". Dirigidos por las computadoras, unos diseñadores automáticos dibujan mediante el sistema de punteado los mapas del tiempo y de los vientos tal como se estaban desarrollando unos minutos antes a cientos de miles de kilómetros. Para enseñarles a dibujar, los ingenieros debieron programar a las máquinas mediante códigos especiales siguiendo siempre el sistema binario.

T A M A Ñ O Y V E L O C I D A D

A medida que fueron desarrollándose las computadoras electrónicas, la preocupación de los científicos se dirigió a lograr mayor velocidad y a obtener menor tamaño de las máquinas. Hoy prácticamente se ha chocado con el límite del tiempo y el espacio. Para comprenderlo quizá convenga reducirlo al absurdo: ¿es posible acortar el tiempo a tal punto que la respuesta sea formulada antes que la pregunta o que los cerebros sean tan pequeños que no se vean a simple vista?

Es que el progreso resultó tan asombroso que prácticamente se ha llegado a una situación límite, en el futuro superada de alguna forma que hoy no podemos imaginar. En 1946, el ENIAC, primer computador electrónico de la Universidad de Pensylvania, realizó una suma en 1/5000 de segundos; pesaba 30 toneladas y ocupaba una superficie de 140 metros cuadrados. Los descendientes de esa criatura realizan hoy la misma operación en 1,5 millonésimas de segundo y podrían colocarse cómodamente en la cocina de cualquier apartamento pequeño. La energía que hubiera necesitado el ENIAC para poder producir tal como lo hacen las modernas máquinas, sólo la podrían haber suministrado la potencia de las cataratas del Niágara. En cambio, los actuales cerebros electrónicos consumen menos energía que la requerida por una pulga.

Las poderosas computadoras que están en funcionamiento tienen el tamaño de la mano de un hombre y pueden emplearse como piloto para un cohete, satélite o nave dirigida.

El tubo de vacío, que sirve para aumentar las señales eléctricas más débiles con gran fidelidad, fue la clave de todas las maravillas de la electrónica, desde el radar hasta la televisión y las computadoras. Quedó suplantado por los transistores, esas piezas diminutas con el mismo poder de amplificación que aquellos, pero que permiten construir radios o cerebros electrónicos mucho más pequeños. Mas la curiosidad investigadora de los hombres no resultó satisfecha con ese hallazgo; hoy las computadoras han sufrido cambios radicales debido al descubrimiento de "bloques de cristal monolítico".

Se trata de piezas microelectrónicas que tienen una estructura molecular capaz de modificarse para transmitir o transmutar la corriente a fin de impulsarla por nuevos caminos. Un sólo cristal, del tamaño de una uña, remplaza una docena de válvulas de vacío y varios metros de cable. A partir de este descubrimiento, los cerebros electrónicos se redujeron en tamaño y aumentaron velocidad en la realización de sus cálculos.

Suponer que existe el límite antes mencionado equivaldría a subestimar la capacidad creativa de los científicos y olvidar los progresos realizados en el campo de la electrónica durante los últimos años.

M A Q U I N A S " I N T E L I G E N T E S "

No es difícil pasar del campo de la electrónica al terreno de la ficción; pero tampoco es sencillo apartarse de este último mientras los avances superan en la práctica las especulaciones de las mentes proclives a la fantasía.

Cuando computadoras de la Unión Soviética traducen jeroglíficos incomprensibles para el hombre, escriben música o componen poemas; cuando cerebros diminutos dirigen y programan la producción de poderosas industrias norteamericanas, cuando todo eso sucede, es comprensible que la mente tarde en asimilar los avances de una disciplina que ha revolucionado al mundo y que seguramente deparará nuevos asombros.

La ciencia electrónica está conduciendo a dos caminos diferentes: uno de ellos se preocupa por producir máquinas que sean cada vez más veloces y más especializadas. El otro se dirige a crear cerebros que se parezcan más y más al hombre. Este último es, sin duda, el más inquietante.

En la actualidad, está tratándose de diseñar "computadores biológicos" que puedan comprender la voz humana y tengan la adaptabilidad de un animal vivo. Las dificultades de esta investigación consiste en que los diversos tipos de sonido emitidos por los hombres son tan diferentes entre sí, que la computadora no puede registrarlos. Se tiende entonces a eliminar los sonidos peculiares de la voz y retener exclusivamente la pronunciación fonética.

No obstante, se ha logrado construir un diseño experimental IBM llamado Shocbox, que reconoce hasta 16 palabras habladas, incluyendo 10 dígitos y los términos de clave aritmética tales como "más", "menos" y "total". Cuando se le ordena, el cerebro electrónico transmite problemas sencillos a una sumadora y la instruye para que los resuelva.

Es probable que dentro de algunos años se logren respuestas habladas de las máquinas, mediante el sistema de grabación del vocabulario que el computador seleccionará electrónicamente para dar la respuesta exacta.

Pero los temores de que la electrónica pueda producir seres capaces de dominar finalmente a sus creadores, escapa a todo razonamiento científico. Durante miles de años las máquinas, desde las primitivas hasta las más recientes, han sido un complemento insoslayable en el desarrollo humano. La necesidad de derivarles a ellas todo aquello que nos es ingrato, pesado o simplemente aburrido, no sólo resulta comprensible sino también saludable.

El surgimiento de la electrónica ha representado una nueva revolución industrial, por lo que de ella dependen todas las ciencias y gran parte del bienestar de los habitantes del planeta.

En cuanto a la competencia, todavía no hay nadie (ni nada) para hacérsela al hombre.

M E D I O S D E I N F O R M A C I O N

ENTRADA	ALMACENAMIENTO	SALIDA
- Lectoras de tarjetas perforadas	- Memoria principal	- Perforadoras de tarjetas
- Lectoras de cintas de papel perforado	- Discos magnéticos	- Perforadoras de cinta de papel
- Lectora de cinta magnética	- Cintas magnéticas	- Cintas magnéticas
- Lectora óptica de caracteres	- Tambores magnéticos	- Discos y tambores magnéticos
- Lectora de cassettes	- Cassettes	- Cassettes
- Lectora de diskettes	- Diskettes	- Diskettes
- Discos y tambores magnéticos		- Impresoras
- Terminales de video		- Graficadoras .
		- Terminales de video

FUNCIONAMIENTO DE UNA COMPUTADORA

Las facultades que el hombre ha otorgado a las computadoras, ha sido el factor principal para que éstas sean consideradas como criaturas, cuyos poderes parecen en condiciones de resolverlo todo. Sin embargo, no son más que "máquinas de alta velocidad capaces de admitir y almacenar datos e instrucciones, procesar o tratar aquéllos de acuerdo con éstas últimas y producir los resultados de esta elaboración en un formato útil y automático". (*)
Esta serie de datos e instrucciones le deben ser dados por el hombre, ya que la computadora por sí misma no tiene capacidad de decisión o de actuación, dependiendo ésta de la inteligencia y habilidad del ser humano.

Toda computadora está compuesta de una parte física llamada HARDWARE y otra lógica conocida como SOFTWARE.

Hardware son los equipos electrónicos, mecánicos y electromecánicos que forman la estructura de la computadora. Esta parte se encarga de captar la información, de las operaciones aritméticas y lógicas, del almacenamiento de la información y de imprimir los resultados. Asimismo, está compuesta de:

a) ELEMENTOS DE ENTRADA, o equipos periféricos, encargados de la captación de datos; por ejemplo lectora de tarjetas.

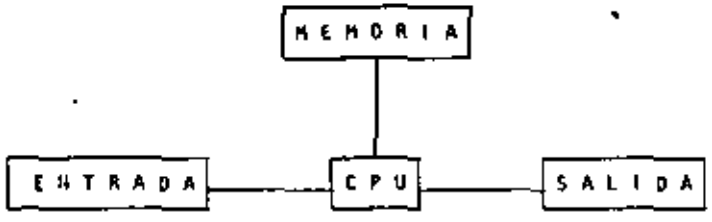
b) PROCESADOR CENTRAL o CPU, en donde residen las unidades de operación aritmética y lógica.

c) DISPOSITIVO DE ALMACENAMIENTO o memoria, donde se guarda la información traducida a números, tanto el problema en sí como la información generada en el curso de las operaciones de cálculo. Para ello se utiliza un conjunto de bits (dígitos binarios), que son la mínima unidad de almacenamiento que puede ser direccionable.

d) ELEMENTOS DE SALIDA, al igual que los de entrada también son dispositivos periféricos, encargados de la impresión de resultados; por ejemplo, las impresoras.

Cabe señalar que existen teletipos y terminales de video, que son pequeñas máquinas mediante las cuales es posible establecer una comunicación directa (vía línea telefónica) con el equipo central, funcionando como elementos de entrada y salida, que permiten procesos conversacionales y el desarrollo de sistemas en tiempo real. Algunas máquinas controlan este tipo de dispositivos a través de un procesador de comunicación de datos.

Representación gráfica del HARDWARE:



El Software, la otra parte de una computadora, está formado por los programas escritos en un lenguaje apropiado a la estructura física de las máquinas, y con los cuales es posible utilizarlas.

Básicamente se tienen:

a) SISTEMA OPERATIVO. Programa almacenado en memoria que se encarga de controlar la asignación del procesador y coordinar las funciones del Hardware; este programa sirve para repartir los recursos de la máquina en forma óptima.

b) COMPILADORES. Programas que generan un grupo de instrucciones-máquina (código que puede ejecutar la computadora) a partir de un programa escrito. Este conjunto de instrucciones es llamado programa objeto y puede ser ejecutado cuantas veces se desee.

c) INTRINSECOS. Pequeños módulos de programa que pueden ser utilizados por diferentes usuarios, sin que ellos tengan necesidad de programarlos; por ejemplo, la raíz cuadrada, funciones trigonométricas, etcétera.

d) INTERPRETE. Programas que traducen instrucciones-máquina, ejecutando cada instrucción traducida sin generar el programa objeto.

e) RUTINAS DE UTILERIA Y PAQUETES DE BIBLIOTECA. Programas especializados que simplifican procesos que comúnmente se llevan a cabo; por ejemplo SPSS (paquete estadístico), paquetes de bases de datos en las diferentes máquinas, etcétera.

Para el correcto funcionamiento de una computadora en una aplicación específica, se debe efectuar un análisis del problema a resolver, reuniéndose el posible usuario de la máquina y la persona que trabaja con la computadora. Una vez realizado el análisis y definido que el mejor medio para resolver el problema es la computadora, se buscará el método más apropiado para hacerlo. Esto significa establecer el funcionamiento lógico y matemático que se seguirá, plasmándolo en un diagrama de flujo o de proceso.

Posteriormente, se define el lenguaje en el que va a programarse, se efectúa el programa, se realiza una prueba con datos conocidos y se hace un estudio comparativo entre los resultados obtenidos con el uso de la computadora y los resultados esperados; si éstos concuerdan, el proceso habrá terminado, si no, deberán corregirse los errores.

(*) Bernice, Daniel D.: Introducción a las Computadoras y Procesos de Datos. 1973.

E L U N I V E R S O D E L A S C O M P U T A D O R A S

Por: Lic. Marcia de las Fuentes

(Tomado de la revista Geografía Universal, Año 7, Vol. 13, No. 6)

Mientras el público observaba con curiosidad, Bouchon colocó la hoja de papel con pequeños agujeros en el rodillo del telar. El cilindro comenzó a girar y la gente lanzó una exclamación de asombro: como por arte de magia, de la máquina fue surgiendo un hermoso tejido perfectamente diseñado en seda. La demostración había tenido buen éxito.

Basile Bouchon fue quien construyó en 1725 el primer telar que podía tejer siguiendo la clave cifrada en una hoja de papel perforado. El método era sencillo y práctico; consistía en realizar agujeros en un rollo de papel siguiendo el diseño que se deseaba tejer. Cuando este papel se presionaba contra una hilera de agujas, las que coincidían con los agujeros permanecían en la misma posición; las otras se movían hacia adelante. De tal forma iba lográndose el dibujo en el tejido. De esta manera nacía el primer "diálogo" entre la máquina y el hombre; comunicación que habría de convertirse, siglos más tarde, en un fundamento de la ciencia y permitiría enormes avances para la tecnología.

La historia de los cerebros electrónicos es muy reciente, aunque sus primitivos antecesores hayan sido creados hace muchos siglos.

Es probable que fuese en Babilonia, 5,000 años atrás, cuando el proceso de contar con los dedos sufrió su primera modificación. Seguramente algún anónimo comerciante, confundido con su dinero, creó el ábaco, ese rudimentario pero efectivo sistema de contabilización. Transmitido a través de todas las civilizaciones, el ábaco se convirtió en el tradicional instrumento que introduce a los niños en el complejo mundo de los números. Los chinos desarrollaron y refinaron ese calculador, y a tal punto lo hicieron, que aún existen algunos profesionales capaces de realizar cuentas con más velocidad que las máquinas de sumar mecánicas.

Pero el ábaco, aunque mucho más efectivo que el viejo sistema de contar con los dedos, está basado en la memoria visual del individuo que lo acciona. Este debe recordar cuántos son los pequeños discos que ha movido -y si lo olvida, volverlos a contar-, para sumarlos o restarlos con las otras unidades ubicadas en la siguiente barra. Necesita, por lo tanto, concentrarse en la operación y dedicar su atención a ella.

En 1612, el filósofo Blaise Pascal inventó una máquina de sumar y restar que era muy superior a la precedente. Se trataba de una pequeña caja que encerraba en su interior cilindros y engranajes; las ruedas de la parte superior del aparato correspondían a las unidades, decenas, centenas y subsiguientes, y cada rueda registraba de cero a nueve. El invento fue valioso aunque poco práctico por sus características.

Más eficientes resultaron los Rodillos de Napier, fabricados por John Napier y que servían para multiplicar. La nobleza europea del siglo XVII los recibió con mucho entusiasmo debido al pequeño tamaño que facilitaba el traslado. Eran rodillos que contenían los dígitos del 1 al 9, con sus múltiples en columnas debajo de ellos. Al hacer girar los rodillos se podía multiplicar fácilmente y sin demorar mucho tiempo. Otro aporte fundamental lo realizó el inglés Charles Babbage, quien entre 1812 y 1822 ideó y realizó un complicado artefacto que podía calcular y hasta imprimir las tablas matemáticas.

U N N U E V O L E N G U A J E

La automación, palabra derivada de automatización y acuñada por el norteamericano Delmar S. Harder, está dirigida a reducir el esfuerzo y dejar que el control mecánico o electrónico sustituya el control que ejerce el cerebro del hombre. Existen, por supuesto, diversos niveles de mecanización; nadie se asombra cuando una aspiradora recoge en pocos segundos el polvo acumulado en el piso y evita el ejercicio manual del barrido. El hábito en el uso de los numerosos artefactos que la industria provee al hogar, ha generado cierta indiferencia en cuanto a su mecanismo. Según expertos, hacia finales de la actual década, los países desarrollados contarán con casi 80 máquinas de distinto tipo para cada uno de los hogares. Estos aparatos, que podrían ubicarse en una categoría inferior, están dedicados a cumplir diversas tareas domésticas.

Pero existe un nivel superior en la automación; ese nivel se llama electrónica y gracias a ella el desarrollo de las disciplinas humanas ha alcanzado una eficiencia admirable.

Este alto nivel de las máquinas se encuentra en las computadoras digitales, perfectos cerebros que no sólo son capaces de controlar su propio funcionamiento, sino también de dirigir y comunicarse con otras máquinas; recibir la información que le envían y procesarla, alertar sobre las posibles deficiencias y subsanarlas en caso de que se produzcan.

No obstante que tales aparatos son de uso corriente en prácticamente todas las disciplinas científicas y técnicas, la mayoría de las personas no ha asimilado aún el verdadero carácter que poseen y observan con incredulidad y hasta cierta desconfianza a estos robots infalibles, presuntos competidores de los hombres.

Una cosa es apretar un botón y que la licuadora, la lavadora o la máquina de afeitar se pongan en funcionamiento, y otra muy distinta es que exista una comunicación directa con un aparato capaz de responder a los interrogantes planteados por un ser humano. En el segundo caso se establece un diálogo entre el hombre y la máquina, y es precisamente eso lo que provoca ciertas reservas en aquellos que temen ver al mundo dominado por cerebros electrónicos.

Pero saliendo de este terreno sin asidero real, el hecho de que se haya encontrado un método de comunicación entre el hombre y el objeto posibilitó que la ciencia avance a pasos gigantescos.

Para lograr esa relación fue necesario crear un nuevo lenguaje, que pudiera ser comprendido por las computadoras y que a su vez le permitiera al hombre recibir los mensajes que ella envía. Ese nuevo idioma se llama cibernética y es una disciplina dedicada a la comunicación entre hombre y máquina, máquina y hombre y máquina y máquina. La palabra deriva del griego kybernetes, y define los mensajes intercambiables que forman la comunicación recíproca.

El problema residía en encontrar un vocabulario apropiado para la computadora, que le permitiera responder a las preguntas. Bouchon, con su telar de papeles perforados, resultó el promotor de ese lenguaje. Los agujeros o no agujeros que iban encontrando las agujas a su paso fueron los antecedentes de las tarjetas perforadas que hoy se utilizan.

En realidad, se trata de un vocabulario simple basado en el sí o no del sistema binario o de dos bases. El lenguaje de un foco de luz, por ejemplo, consiste en encendido y apagado; de igual forma, las tarjetas que se introducen en la computadora son leídas por ellas mismas casi como en el telar de Bouchon: la corriente eléctrica penetra en el agujero de la ficha o no lo hace.

Puesto que la computadora se limita a la respuesta de si ha penetrado o no, los números que se introducen deberán ser perforados en las fichas mediante el código binario expresado en agujeros y espacios.

Ese es el sistema básico y a partir de él se crearon otros métodos que siguen el mismo procedimiento; uno de ellos es la cintá perforada y el otro la cinta magnetofónica. En este último caso, cada cinta tiene siete canales en los que se encuentran puntos magnetizados con cabezas de electroimanes. Al pasar esos puntos por el mecanismo lector, se convierten en pulsaciones eléctricas que van traduciendo el mensaje. Esto se realiza a una velocidad de 630,000 puntos por segundo.

Las computadoras no sólo han reducido el esfuerzo humano en la industria y la investigación, sino que además han posibilitado una mayor rapidez en los procesos algebraicos y han eliminado el margen de error. La mayoría de cerebros electrónicos está dotada del sistema "feedback". Estas máquinas poseen un autocontrol capaz de corregir sus propias deficiencias y las de aquellas que estén bajo su dirección. En caso de que una comience a funcionar mal, el cerebro principal recibirá de inmediato la señal de alarma y buscará en su memoria cuales son los procedimientos adecuados para corregir el problema. Si está en condiciones de solucionarlo, lo hará sin que tenga que intervenir ningún hombre. En caso contrario detendrá el funcionamiento de la máquina descompuesta y avisará al operador qué sucede y dónde debe dirigirse para encontrar la falla.

E L "P E N S A M I E N T O" E L E C T R O N I C O

La pregunta surge espontáneamente: ¿cómo un aparato compuesto por circuitos eléctricos y sistemas magnéticos puede "pensar" o tener la suficiente autonomía a fin de controlar el funcionamiento de otras máquinas sin la presencia humana?

En primer término, debe recordarse que las computadoras no pueden realizar nada que no haya sido programado previamente por el hombre. Son cerebros electrónicos que sólo se ponen en funcionamiento cuando se les suministran los elementos necesarios para que lo hagan. Y no hay nada que los asemeje a una inteligencia artificial.

Para que esas máquinas cumplan con su tarea, es necesario alimentarlas; se les da el problema y la información que necesitarán a fin de solucionarlo. A partir de ese momento, la información suministrada pasará al sistema de control y al sistema de memoria.

El primero toma la información y la organiza para su posterior selección. El segundo comprueba que todos los datos estén correctos y que no haya error alguno; en caso de que advierta una equivocación, avisará cuál es y dónde está. Por ejemplo, si se ha introducido en la computadora un texto y hay una frase en la cual se abre un signo de paréntesis que luego no cierra, el control dará la alarma. Es que el sistema observó la apertura del signo y mientras continúa recorriendo las palabras siguientes espera la llegada del cierre, porque ha sido programado para que cada vez que se abra un paréntesis se cierre posteriormente; si así no ocurre es porque hay un error.

El sistema de memoria es el que recibe toda la información y la almacena en sus unidades. Le servirá de antecedente cada vez que tenga que retornar al mismo caso. Allí se mantienen todos los datos clasificados y listos para ser utilizados cuando el control los necesite. Existe también la biblioteca, que es el lugar donde se guardan los métodos para solucionar problemas. Por medio de circuitos, este sistema brinda las instrucciones básicas que previamente le han dado los operadores.

El paso siguiente es la solución del problema. En forma distinta que el cerebro humano, la computadora actúa por repetición, con la lógica suminstrada por un programador humano.

Finalmente, la máquina otorga la respuesta de acuerdo con el sistema en que opere: fichas perforadas, cintas magnéticas, hojas escritas a máquina.

La explicación más elemental de una computadora podrá realizarse de la siguiente forma: a) suministro de información (alimentación); b) almacenamiento de esa información (memoria); c) solución al problema (elaboración); d) respuesta final del resultado del problema.

D I V E R S A S F U N C I O N E S

Las necesidades científicas han impuesto una amplia diversificación entre las mismas computadoras, de acuerdo con las funciones que deben cumplir. Existen dos clases de máquinas; la denominada Analog (vocablo derivado del griego análogos) y la Digital, del latín digitus o "dedos", así llamado por la costumbre de contar con los dedos.

Las primeras no se ocupan de los números sino de cantidades físicas análogas; su trabajo consiste en expresarse en términos físicos y no numéricos, como por ejemplo, el ángulo de rotación de un eje, el voltaje eléctrico, etcétera. En cambio, las digitales se dedican a calcular y computar numéricamente; viven de pulsaciones eléctricas que recuerdan al antiguo sistema de Bouchon y son capaces de resolver complicados problemas algebraicos con mucha mayor velocidad que 500,000 hombres trabajando simultáneamente, no con papel y lápiz, sino con calculadoras manuales.

Los ingenieros que programaron los viajes espaciales y el primer descenso del hombre en la Luna, admitieron que todas esas actividades hubieran sido imposibles de realizar si no contaban con las modernas computadoras. Ningún ser humano está en condiciones de calcular las trayectorias, la propulsión y las necesidades de combustible con la suficiente exactitud como para hacer posible el experimento.

Los millones de sumas y restas requeridos para calcular los constantes cambios de gravitación de la Tierra, la Luna y el Sol, hubieran ocupado la actividad de varios miles de ingenieros que, de todos modos, habrían tardado varios siglos en llegar a conclusiones no del todo exactas.

Si bien lo anterior da una idea aproximada de la labor que cumplen estas criaturas electrónicas, no menos significativo es recordar que en la mayoría de los países las tareas de mantenimiento de ser-

vicios están a cargo de computadoras. Son ellas quienes controlan el suministro de electricidad y avisan sobre las reparaciones que deberán hacerse en el futuro, las que regulan el uso de agua potable en las ciudades o informan a los fabricantes de automóviles acerca de las tendencias del mercado y de las necesidades que se deberán tomar en cuenta. Aconsejan a los productores de manzanas de Nueva Inglaterra sobre los periodos de cosecha más óptimos, recopilan fórmulas de mezclas para fabricantes de piensos de ganado vacuno y de aves o ayudan a los médicos a determinar las dosis de radiación para los enfermos de cáncer.

La meteorología, considerada siempre como un arte de la predicción con un alto margen de error, ha logrado progresos notables gracias al funcionamiento de estos cerebros. Antes de constituirse en ciencia, estaba basada en las impresiones personales de los campesinos, el comportamiento de los pájaros o el dolor reumático de alguna anciana. A través de ellos se efectuaban los pronósticos del tiempo y sus posibles variaciones; la lluvia, la humedad, la sequía eran "advertidas" mediante signos que brindaba la propia naturaleza e "interpretados" arbitrariamente por los hombres.

El meteorólogo, científicamente, medía la presión atmosférica, observaba las nubes y hacía algunos pequeños experimentos que le proporcionaban datos, para determinar los cambios futuros de clima. Hoy existen computadoras electrónicas que pueden efectuar un millón de cálculos por segundo y que son capaces de anunciar las lluvias o las sequías conforme al análisis de información suministrada.

El Centro Meteorológico Nacional -NMC- de los Estados Unidos, recibe cuatro veces por día los datos enviados por varios satélites en órbita, así como por 2,000 estaciones meteorológicas de todo el mundo, más 3,200 informes de aviones comerciales y alrededor de 200 reportes elaborados (por computadoras) en vuelos de reconocimiento. Sería imposible que ese caudal de datos fuera recopilado, estudiado e interpretado por los hombres; en cambio, un enorme complejo de computadoras se encarga de hacerlo. Devora miles de informes sobre el tiempo, los selecciona y realiza millones de cálculos en un lapso de 90 minutos; luego saca conclu siones y las entrega. Actualmente, las predicciones para las 18 horas siguientes se consideran exactas en un 85 por ciento y se calcula que con las pequeñas computadoras que ya están circulando en los satélites alrededor de la Tierra y suministrando más información, en los próximos años se obtendrán conocimientos suficientes como para hacer predicciones absolutamente ciertas.

En este caso, los cerebros electrónicos no se limitan a suministrar los datos sino también "graficarlos". Dirigidos por las computadoras, unos diseñadores automáticos dibujan mediante el sistema de punteado los mapas del tiempo y de los vientos tal como se estaban desarrollando unos minutos antes a cientos de miles de kilómetros. Para enseñarles a dibujar, los ingenieros debieron programar a las máquinas mediante códigos especiales siguiendo siempre el sistema binario.

T A M A Ñ O Y V E L O C I D A D

A medida que fueron desarrollándose las computadoras electrónicas, la preocupación de los científicos se dirigió a lograr mayor velocidad y a obtener menor tamaño de las máquinas. Hoy prácticamente se ha chocado con el límite del tiempo y el espacio. Para comprenderlo quizá convenga reducirlo al absurdo: ¿es posible acortar el tiempo a tal punto que la respuesta sea formulada antes que la pregunta o que los cerebros sean tan pequeños que no se vean a simple vista?

Es que el progreso resultó tan asombroso que prácticamente se ha llegado a una situación límite, en el futuro superada de alguna forma que hoy no podemos imaginar. En 1946, el ENIAC, primer computador electrónico de la Universidad de Pensylvania, realizó una suma en 1/5000 de segundos; pesaba 30 toneladas y ocupaba una superficie de 140 metros cuadrados. Los descendientes de esa criatura realizan hoy la misma operación en 1,5 millonésimas de segundo y podrían colocarse cómodamente en la cocina de cualquier apartamento pequeño. La energía que hubiera necesitado el ENIAC para poder producir tal como lo hacen las modernas máquinas, sólo la podrían haber suministrado la potencia de las cataratas del Niágara. En cambio, los actuales cerebros electrónicos consumen menos energía que la requerida por una pulga.

Las poderosas computadoras que están en funcionamiento tienen el tamaño de la mano de un hombre y pueden emplearse como piloto para un cohete, satélite o nave dirigida.

El tubo de vacío, que sirve para aumentar las señales eléctricas más débiles con gran fidelidad, fue la clave de todas las maravillas de la electrónica, desde el radar hasta la televisión y las computadoras. Quedó suplantado por los transistores, esas piezas diminutas con el mismo poder de amplificación que aquellos, pero que permiten construir radios o cerebros electrónicos mucho más pequeños. Mas la curiosidad investigadora de los hombres no resultó satisfecha con ese hallazgo; hoy las computadoras han sufrido cambios radicales debido al descubrimiento de "bloques de cristal monolítico".

Se trata de piezas microelectrónicas que tienen una estructura molecular capaz de modificarse para transmitir o transmutar la corriente a fin de impulsarla por nuevos caminos. Un sólo cristal, del tamaño de una uña, remplaza una docena de válvulas de vacío y varios metros de cable. A partir de este descubrimiento, los cerebros electrónicos se redujeron en tamaño y aumentaron velocidad en la realización de sus cálculos.

Suponer que existe el límite antes mencionado equivaldría a subestimar la capacidad creativa de los científicos y olvidar los progresos realizados en el campo de la electrónica durante los últimos años.

M A Q U I N A S " I N T E L I G E N T E S "

No es difícil pasar del campo de la electrónica al terreno de la ficción; pero tampoco es sencillo apartarse de este último mientras los avances superan en la práctica las especulaciones de las mentes proclives a la fantasía.

Cuando computadoras de la Unión Soviética traducen jeroglíficos incomprensibles para el hombre, escriben música o componen poemas; cuando cerebros diminutos dirigen y programan la producción de poderosas industrias norteamericanas, cuando todo eso sucede, es comprensible que la mente tarde en asimilar los avances de una disciplina que ha revolucionado al mundo y que seguramente deparará nuevos asombros.

La ciencia electrónica está conduciendo a dos caminos diferentes: uno de ellos se preocupa por producir máquinas que sean cada vez más veloces y más especializadas. El otro se dirige a crear cerebros que se parezcan más y más al hombre. Este último es, sin duda, el más inquietante.

En la actualidad, está tratándose de diseñar "computadores biológicos" que puedan comprender la voz humana y tengan la adaptabilidad de un animal vivo. Las dificultades de esta investigación consiste en que los diversos tipos de sonido emitidos por los hombres son tan diferentes entre sí, que la computadora no puede registrarlos. Se tiende entonces a eliminar los sonidos peculiares de la voz y retener exclusivamente la pronunciación fonética.

No obstante, se ha logrado construir un diseño experimental IBM llamado Shocbox, que reconoce hasta 16 palabras halladas, incluyendo 10 dígitos y los términos de clave aritmética tales como "más", "menos" y "total". Cuando se le ordena, el cerebro electrónico transmite problemas sencillos a una sumadora y la instruye para que los resuelva.

Es probable que dentro de algunos años se logren respuestas habladas de las máquinas, mediante el sistema de grabación del vocabulario que el computador seleccionará electrónicamente para dar la respuesta exacta.

Pero los temores de que la electrónica pueda producir seres capaces de dominar finalmente a sus creadores, escapa a todo razonamiento científico. Durante miles de años las máquinas, desde las primitivas hasta las más recientes, han sido un complemento insoslayable en el desarrollo humano. La necesidad de derivarles a ellas todo aquello que nos es ingrato, pesado o simplemente aburrido, no sólo resulta comprensible sino también saludable.

El surgimiento de la electrónica ha representado una nueva revolución industrial, por lo que de ella dependen todas las ciencias y gran parte del bienestar de los habitantes del planeta.

En cuanto a la competencia, todavía no hay nadie (ni nada) para hacérsela al hombre.

M E D I O S D E I N F O R M A C I O N

ENTRADA	ALMACENAMIENTO	SALIDA
- Lectoras de tarjetas perforadas	- Memoria principal	- Perforadoras de tarjetas
- Lectoras de cintas de papel perforado	- Discos magnéticos	- Perforadoras de cinta de papel
- Lectora de cinta magnética	- Cintas magnéticas	- Cintas magnéticas
- Lectora óptica de caracteres	- Tambores magnéticos	- Discos y tambores magnéticos
- Lectora de cassettes	- Cassettes	- Cassettes
- Lectora de diskettes	- Diskettes	- Diskettes
- Discos y tambores magnéticos		- Impresoras
- Terminales de video		- Graficadoras
		- Terminales de video

FUNCIONAMIENTO DE UNA COMPUTADORA

Las facultades que el hombre ha otorgado a las computadoras, ha sido el factor principal para que éstas sean consideradas como criaturas, cuyos poderes parecen en condiciones de resolverlo todo. Sin embargo, no son más que "máquinas de alta velocidad capaces de admitir y almacenar datos e instrucciones, procesar o tratar aquéllos de acuerdo con éstas últimas y producir los resultados de esta elaboración en un formato útil y automático". (*) Esta serie de datos e instrucciones le deben ser dadas por el hombre, ya que la computadora por sí misma no tiene capacidad de decisión o de actuación, dependiendo ésta de la inteligencia y habilidad del ser humano.

Toda computadora está compuesta de una parte física llamada HARDWARE y otra lógica conocida como SOFTWARE.

Hardware son los equipos electrónicos, mecánicos y electromecánicos que forman la estructura de la computadora. Esta parte se encarga de captar la información, de las operaciones aritméticas y lógicas, del almacenamiento de la información y de imprimir los resultados. Asimismo, está compuesta de:

a) ELEMENTOS DE ENTRADA, o equipos periféricos, encargados de la captación de datos; por ejemplo lectora de tarjetas.

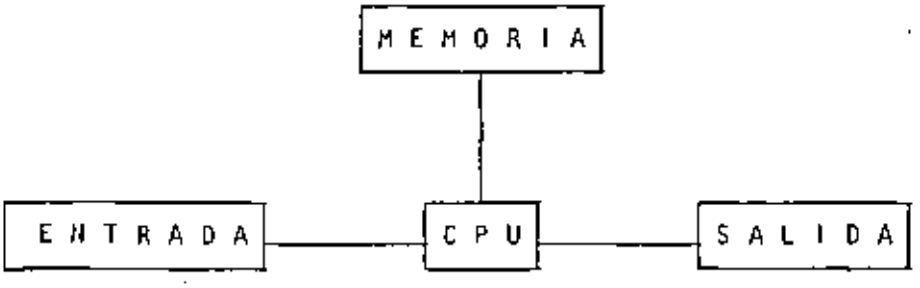
b) PROCESADOR CENTRAL o CPU, en donde residen las unidades de operación aritmética y lógica.

c) DISPOSITIVO DE ALMACENAMIENTO o memoria, donde se guarda la información traducida a números, tanto el problema en sí como la información generada en el curso de las operaciones de cálculo. Para ello se utiliza un conjunto de bits (dígitos binarios), que son la mínima unidad de almacenamiento que puede ser direccionable..

d) ELEMENTOS DE SALIDA, al igual que los de entrada también son dispositivos periféricos, encargados de la impresión de resultados; por ejemplo, las impresoras.

Cabe señalar que existen teletipos y terminales de video, que son pequeñas máquinas mediante las cuales es posible establecer una comunicación directa (vía línea telefónica) con el equipo central, funcionando como elementos de entrada y salida, que permiten procesos conversacionales y el desarrollo de sistemas en tiempo real. Algunas máquinas controlan este tipo de dispositivos a través de un procesador de comunicación de datos.

Representación gráfica del HARDWARE:



El Software, la otra parte de una computadora, está formado por los programas escritos en un lenguaje apropiado a la estructura física de las máquinas, y con los cuales es posible utilizarlas.

Básicamente se tienen:

a) SISTEMA OPERATIVO. Programa almacenado en memoria que se encarga de controlar la asignación del procesador y coordinar las funciones del Hardware; este programa sirve para repartir los recursos de la máquina en forma óptima.

b) COMPILADORES. Programas que generan un grupo de instrucciones-máquina (código que puede ejecutar la computadora) a partir de un programa escrito. Este conjunto de instrucciones es llamado programa objeto y puede ser ejecutado cuantas veces se desee.

c) INTRINSECOS. Pequeños módulos de programa que pueden ser utilizados por diferentes usuarios, sin que ellos tengan necesidad de programarlos; por ejemplo, la raíz cuadrada, funciones trigonométricas, etcétera.

d) INTERPRETE. Programas que traducen instrucciones-máquina, ejecutando cada instrucción traducida sin generar el programa objeto.

e) RUTINAS DE UTILERIA Y PAQUETES DE BIBLIOTECA. Programas especializados que simplifican procesos que comúnmente se llevan a cabo; por ejemplo SPSS (paquete estadístico), paquetes de bases de datos en las diferentes máquinas, etcétera.

Para el correcto funcionamiento de una computadora en una aplicación específica, se debe efectuar un análisis del problema a resolver, reuniéndose el posible usuario de la máquina y la persona que trabaja con la computadora. Una vez realizado el análisis y definido que el mejor medio para resolver el problema es la computadora, se buscará el método más apropiado para hacerlo. Esto significa establecer el funcionamiento lógico y matemático que se seguirá, plasmándolo en un diagrama de flujo o de proceso.

Posteriormente, se define el lenguaje en el que va a programarse, se efectúa el programa, se realiza una prueba con datos conocidos y se hace un estudio comparativo entre los resultados obtenidos con el uso de la computadora y los resultados esperados; si éstos concuerdan, el proceso habrá terminado, si no, deberán corregirse los errores.

(*) Bernice, Daniel D.: Introducción a las Computadoras y Procesos de Datos. 1973.

DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA U.N.A.M.

LENGUAJE DE PROGRAMACION BASIC - CON APLICACIONES
(PRIMERA PARTE)

TRS - 80 MODEL III
MICROCOMPUTER SISYEM

FEBRERO, 1983

Palacio de Minería Calle de Tacuba 5 primer piso Deleg. Cuauhtemoc 06000 México, D.F. Tel.: 521-40-20 Apdo. Postal M-2285

TRS - 80 MODEL III
MICROCOMPUTER SYSTEM

GUIA DE REFERENCIA

CONTENIDO.

TRS - 80 MODEL III
MICROCOMPUTER SYSTEM

GUIA DE REFERENCIA

I. ENCENDIDO.

Al iniciar, todos los periféricos y la computadora deberán estar apagados.

1. Primero encender los periféricos (impresora normalmente) y después encender la computadora.

2. Deberá aparecer el mensaje.

CASS?

Para seleccionar alta velocidad de cassette (1500 Bits/seg), se presiona H o ENTER. Normalmente se usa esta velocidad.

Para una velocidad de transmisión baja (500 Bits/seg) se debe presionar L , velocidad usada para salvar o cargar programas BASIC nivel II.

3. Después aparece el mensaje

MEMORY SIZE?

Esto nos permite reservar memoria, para utilizar la totalidad de ella sólo se presiona ENTER.

Para reservar memoria, se teclea en decimal la mayor dirección de memoria que se desea usar y se termina con ENTER.

4. Finalmente deberá aparecer el mensaje

```
MODEL III BASIC

(c) TANDY '80

READY
```

Y la computadora estará lista para su uso.

II. PROPOSICIONES Y DECLARACIONES.

Las siguientes proposiciones pueden ser utilizadas tanto en modo directo (sin etiqueta) como en modo ejecución (con etiqueta y por medio de la instrucción RUN).

Todas las proposiciones se encuentran referidas a la página en que se encuentran en el manual TRS-80 MODEL III operation and BASIC Language Reference Manual.

- AUTO inicio, incremento Pag. 125

Nos da una secuencia automática de líneas numeradas.

AUTO inicia en 10 y se incrementa en 10
AUTO5,5 inicia en 5 y se incrementa en 5
AUTO,5 inicia en 0 y se incrementa en 5
AUTO50, inicia en 50 y se incrementa en 10

- CLEAR n Pag. 126

Inicializa todas las variables a cero (únicamente CLEAR). Con argumento reserva n bytes a todas las variables de tipo string.

CLEAR 100 reserva 100 bytes.

Se debe tener en cuenta que al ejecutar esta instrucción las variables declaradas con DEFSTR regresan a su tipo original.

- CLOAR "nombre" Pags. 29-34, 126.

Carga a la computadora un programa BASIC que se encuentra en cinta; al utilizarlo deberá estar conectada a la computadora la casettera con el cassette en que se halla el programa montado por el lado debido, y con la tecla "PLAY" oprimida.

Sólo es usado el primer caracter del nombre.

CLOAD Carga el primer programa que se encuentre.

CLOAD"PRO" Carga el primer programa que halla sido grabado con un nombre iniciado con P.

CLOAD?"P" Compara el programa "P" en cinta con el programa que en ese momento reside en la computadora, byte por byte.

Nos sirve para verificar una correcta transferencia después de salvar un programa en cinta. Pag. 127.

- CLS Pag. 186.

Borra todo lo que hay en la pantalla y sitúa al cursor en la esquina superior izquierda.

- CONT Pag. 127.

Continúa la ejecución de un programa que ha sido interrumpido con la instrucción STOP o la tecla BREAK.

La ejecución continúa a partir del estado que se tenía al suspender el programa, a menos que se haya modificado alguna variable en modo de ejecución directa.

- CSVE "nombre" Pag. 29-34, 127.

Guarda un programa residente en cinta. El nombre puede formarse por letras, números o caracteres especiales (excepto comillas), sin embargo, solo se tomará encuenta el primer caracter encontrado.

CSAVE "I/HOLA" Guarda el programa residente en cinta, con el nombre " I "

- DATA Pag. 142.

Almacena datos que serán accesados por medio de una proposición READ. Los elementos del DATA serán leídos secuencialmente a partir de la primera proposición data. Los datos de tipo string no necesitan encerrarse entre comillas a menos que contengan blancos o comas.

DATA UN DATO, "Y/O ESTE",123

- DEFDBL variables, rango Pag. 149.

Define variables de doble precisión que permite 17 dígitos de precisión. Si se define un rango, vgr. A-F, todas las variables que comiencen con una letra de ese rango serán de doble precisión.

DEFDBL A-F, X

- DEFINT variables, rango Pag. 148.

Define variables de tipo entero. Ayuda a ahorrar memoria, puesto que los valores enteros ocupan menos espacio de ella. Todas las variables que comiencen con una letra definida entera, serán de tipo entero.

DEFINT I-N

- DEFSNG variable, rango Pag. 148.

Define variables de tipo simple precisión, esto es, solo se guardarán 7 dígitos en memoria. Funciona como DEFDBL y DEFINT, y al igual que en ellas se debe notar que aunque alguna letra este definida con ese tipo, éste puede ser cambiado con los caracteres de declaración de tipo.

DEFSNG A-F Define el rango A a F como simple precisión.

Sin embargo ALFA# será tratada como variable de doble precisión.

- DEFSTR variable, rango Pag. 148.

Define variables de tipo "string". Normalmente las variables de tipo "string" pueden contener hasta 50 caracteres; esto se puede cambiar con la proposición CLEAR n.

DEFSTR N,O-F

- DELETE número de secuencia o rango Pag. 128.

Elimina de memoria las líneas de programa especificadas. Se puede referir a una línea o a una secuencia de líneas.

DELETE 80 Borra de memoria la línea 80

DELETE-80 Borra desde el principio del programa hasta la línea 80 inclusive.

DELETE 10-80 Borra de la línea 10 a la 80.

DELETE Borra la última línea editada o grabada en memoria.

- DIM Var (dimensión) Pag. 150-151,173-178.

Dimensiona uno o más arreglos; la dimensión de cada arreglo será de 0 al número que se especifique.

DIM A,B(3),D(2,2) Reserva memoria para los arreglos A y B que tendrán de 0 a 3 suscriptores, esto es, A(0), A(1), A(2), A(3). La variable D será una matriz de 3 x 3 elementos.

Cuando se omite la dimensión de alguna variable, ésta tiene 11 elementos (0 a 10).

También se pueden dimensionar variables de tipo "string".

Si en algún lugar de un programa se redimensiona una variable, primero se deberá ejecutar la instrucción CLEAR, de lo contrario se producirá un error por dimensionar dos veces la misma variable.

• EDIT número de línea Pag. 14,128,195-201.

Cambia la computadora a modo de edición. En este modo se puede corregir o alterar líneas del programa residente. Consultar comandos de edición.

EDIT 100 La línea 100 aparecerá para ser editada.

EDIT La última línea referenciada aparece para ser editada.

• END Pag. 151.

Termina normalmente la ejecución de un programa, se utiliza principalmente para provocar que la ejecución termine en un lugar del programa que no es precisamente la última línea de él.

• ERROR(n) Pag. 158.

Simula el error referido con el número *n* como parámetro. Su principal uso es en la prueba de rutinas de error (vease ON ERROR GOTO), la computadora procede exactamente como si ese error hubiera ocurrido (consultar la tabla de errores. (Pag. 223-225).

ERROR(2) Simula un error de sintaxis.

• FOR - TO - STEP - ... NEXT Pag. 155-157.

Provoca la iteración de las instrucciones que se encuentran entre FOR-TO-STEP y NEXT tantas veces como lo permita el límite. Al ejecutarse una instrucción de este tipo primero se realiza lo que esté antes del NEXT, al llegar ahí se incrementa el contador y se compara con el límite; en caso de ser mayor a él, continúa con la siguiente instrucción, de lo contrario se regresa a la primera después de FOR. Si se omite el incremento "STEP", se toma como 1. Es posible utilizar incrementos fraccionales y negativos.

```
FOR I=1 TO 10 STEP 0.5
    PRINT I;
NEXT I
```

• GOSUB número de línea Pag. 153.

Transfiere el control del programa a la línea especificada a partir de dicha línea en adelante, se considerará como una subrutina hasta encontrar la proposición RETURN que regresará el mando a la siguiente instrucción después del GOSUB.

GOSUB 1000 Transfiere el control a la línea 1000.

- GOTO número de línea Pag. 152.

Transfiere el control del programa a la línea especificada. El programa continúa su ejecución normal; en modo inmediato, se puede usar para iniciar la corrida de un programa en una línea determinada.

GOTO 1000 Transfiere el control a la línea 1000.

- IF - THEN - ELSE Pag. 160-161.

Prueba una expresión lógica o relacional y:

Si es verdadera ejecuta la instrucción o instrucciones que se encuentran entre THEN y ELSE.

Si es falsa ejecuta la ó las instrucciones que están después de ELSE.

```
IF A$="SI" THEN PRINT"CONTINUA":PRINT"ARRANCA"
          ELSE PRINT"FINALIZA":END
```

En el caso de que A$ sea igual a la cadena "SI", se imprimirá CONTINUA y ARRANCA, de lo contrario imprimirá FINALIZA y dará por terminado el programa.

Al utilizar el ELSE se debe cuidar continuar en la misma línea lógica (sin oprimir ENTER).

Es posible omitir la opción ELSE. En este caso, al ser verdadera la expresión evaluada se ejecuta la proposición indicada (en caso de tener varias instrucciones separadas por dos puntos en una misma línea se ejecutan todas ellas) y cuando la expresión es falsa, se produce un brinco a la siguiente línea lógica.

IF P=Q THEN 200 De ser verdadera se irá a la línea 200.

- INPUT Lista de variables Pag. 140-141.

Causa la suspensión del programa hasta que se hayan dado por medio del teclado los valores de las variables especificadas en la lista. La lista de variables puede incluir valores numéricos y cuerdas. Al teclear los datos, estos deberán ir separados por comas, y en caso de cuerdas que contengan blancos, comas o puntos, se deberán encerrar entre comillas.

INPUT A,A$ Pedirá el valor numérico de A y la cuerda A$.

Como ayuda, es posible desplegar un mensaje en la pantalla al solicitar un INPUT.

```
INPUT"DAME TU NOMBRE"; NO$
```

- INPUT #-1 lista de variables Pag. 145.

Se utiliza para leer valores guardados en un cassette. Los valores deben coincidir en cantidad, orden y tipo con los que son solicitados en el INPUT. Se debe situar la cinta al inicio de los valores a leer y tener la grabadora con el "PLAY" encendido.

- LET asignación. Pag. 151.

Se utilizaba en BASIC standard; no es necesario para esta máquina.

```
LET A = B*C
```

- LIST secuencia. Pag. 13, 31, 35, 128.

Causa la visualización en pantalla de la línea o líneas que se le indique.

LIST 50	Aparece la línea 50
LIST 50-100	Aparece de la línea 50 a la 100
LIST 50-	Aparece de la línea 50 al fin del programa.
LIST.	Aparece la última línea referida.

- LLIST secuencia Pag. 129.

Actúa como LIST, solo que la visualización se hace en la impresora de papel.

- LPRINT lista de variables Pag. 144.

Actúa como PRINT, solo que la impresión es a papel.

LPRINT TAB(t) Pag. 144.

Funciona como PRINT TAB, solo que la impresión es a papel.

- LPRINT USING

Similar a PRINT USING, impresión a papel.

- NEW Pag. 129.

Borra el programa residente, inicializa todas las variables a cero (nulo). Se utiliza para iniciar un nuevo programa.

- ON ERROR GOTO número de línea Pag. 158.

Cuando la computadora encuentra un error suspende el programa, a menos de que una instrucción ON ERROR haya estado antes de la línea de error; en ese caso la secuencia de programa brincará a la línea que se le indique (normalmente es una rutina de manejo de errores).

```
ON ERROR GOTO 1000
```

Para deshabilitar una rutina de manejo de error, se utiliza lo siguiente: ON ERROR GOTO 0

ON - GOSUB lista de etiquetas Pag. 155.

Es un salto condicional a rutinas del programa. Evalúa una variable o expresión y dependiendo de su valor transfiere el control al número de línea que se encuentra en el lugar correspondiente dentro de la lista.

El resultado de la evaluación se debe encontrar entre 0 y 255, de lo contrario produce error. Si el resultado de la evaluación tiene parte fraccionaria, sólo se tomará la parte entera.

ON N GOSUB 100, 200, 300

SI N = 0 continúa normalmente

N = 1 va a la línea 100

N = 2 va a la línea 200

N = 3 va a la línea 300

Proseguirá a partir de la línea a donde se hizo el salto hasta encontrar la proposición RETURN que regresa el control a la siguiente instrucción al ON GOSUB.

ON - GOTO lista de etiquetas

Salto condicional a líneas simples de programa. Funciona similar al ON-GOSUB, solo que no se puede regresar el control usando RETURN.

ON SNG(X) + 2 GOTO 200, 300, 400

OUT punto, valor Pag. 189.

Envía el valor-byte al puerto especificado. Puerto y valor están en el rango de 0 a 255.

POKE n, v Pag. 189.

Guarda el valor v (entre 0 y 255) en la dirección de memoria n (entre 0 y 32767). De este modo podemos, por ejemplo; hacer que el cursor no "parpadee". Usamos

POKE 16412,1

PRINT lista Pag. 133.

Sirve para visualizar información en la pantalla. La lista puede ser mensajes encerrados entre comillas, variables string, números constantes, variables o expresiones que contengan cualquiera de los elementos anteriores. Los elementos de la lista pueden estar separados por comas o punto y coma; la coma produce un avance a la siguiente zona de impresión.

Si se utiliza punto y coma, se inserta un espacio después de escribir un valor numérico, después de escribir un string no inserta espacios. El punto y como al final de una lista de PRINT suprime el regreso automático del cursor.

La pantalla se divide en cuatro zonas de impresión, teniendo cada una de ellas dieciseis espacios.

```
PRINT "LA RESPUESTA ES:";RS$

PRINT A,B,C;
```

La instrucción PRINT puede tener los siguientes modificadores:

n — Comienza una impresión en la posición n de la pantalla; n debe ser un número entre 0 y 1023. /pag.134/

```
PRINT  509,"CENTRO"
```

TAB(n) — Mueve el cursor a una posición especificada por n. La posición n puede estar dada por una expresión y será hasta 127. Es posible utilizar más de un TAB en una instrucción PRINT. /pag. 135/.

```
PRINT TAB(32)   "HOLA"
PRINT TAB(2*X)  "HOLA;TAB(2*X+5)N$
```


USING — Permite especificar el formato en que se hará una impresión. Tiene la siguiente forma:

```
PRINT USING string; valor(es)
```

En donde string es una cadena con especificadores de campo que define la forma en que se imprimirá lo que precede al punto y coma. /pag. 136 - 140/.

ESPECIFICADORES DE CAMPO COMO COMPLEMENTO DE USING

\# — Para dar formato a números. El número de símbolos # usados especifica el campo que tendrá el número. Si el campo es mayor que el número, las posiciones no usadas del campo se imprimen como espacios a la izquierda, a menos que este después del punto decimal en cuyo caso se imprimirán ceros a la derecha. Cuando el número excede el tamaño del campo, aparecerá completo y precedido por el símbolo %.

. — Indica la posición del punto decimal.

, Colocada en cualquier posición entre el primer dígito y el punto, causa que se imprima una coma a la izquierda de cada 3 dígitos.

** Colocados al principio del campo, provoca que todos los espacios no usados a la izquierda sean llenados por asteriscos.

$ Se imprime un signo de pesos en la primer posición del campo.

$$ Si se colocan dos signos de pesos, el signo de pesos se imprime a la izquierda del primer dígito.

**$ Causa que todas las posiciones no usadas sean llenadas con asteriscos y el signo de pesos se imprime junto al primer dígito.

[[[[Causan que el número sea impreso en forma exponencial (E o D).

+ Cuando se coloca al principio del campo, se imprimirá + para números positivos y para números negativos.

- Colocado al final del campo, produce que se imprima en ese lugar un signo - para números negativos, y un espacio en blanco para números positivos.

% % Especifica un campo para variables o constantes de tipo string y de más de un caracter. La longitud del campo será igual al número de espacios entre los signos % más dos.

! Da campo para el primer caracter del string que se imprima en ese campo.

Cualquier otro caracter que se utilice, simplemente se imprimirá como tal en la posición que ocupe.

Ejemplos:

```
10 A$="###.#"
20 PRINT USING A$;X
   PRINT USING "**$##.#";COSTO
   PRINT USING "##.#[[[[";NUM
   PRINT USING "!.!%      %";N1$,N2$,N3$
```

RANDOM Pag. 182

Da una nueva semilla al generador de números pseudo aleatorios. Se utiliza al principio de un programa cuando deseamos asegurar una secuencia impredecible de números RANDOM.

READ lista Pag. 142

Asigna a los elementos de la lista de variables los valores definidos en una instrucción DATA. La primer instrucción READ usada en el programa asignará los primeros valores de la primer instrucción DATA y los READ posteriores asignarán los siguientes valores encontrados y así sucesivamente.

```
20  READ A,B,N$
```

REM Pag. 160.

Se utiliza para introducir comentarios. Utilizando REM, los siguientes 255 caracteres que se escriban, serán ignorados. Se puede utilizar un apóstrofo para abreviar.

```
10  REM *** INICIO ***

20  X = X[ 2 REM  X CUADRADA

30  Y = X[ 3  ' X CUBICA
```


RESET(x,y) Pag. 186.

Apaga un punto de la pantalla situado por las coordenadas x,y. Ver también SET(x,y).

```
RESET (P,10)
```

RESTORE Pag. 143.

Regresa el apuntador de los elementos de DATA al primer valor del primer DATA en el programa. Se utiliza cuando quieren usar las mismas líneas de DATA ya utilizadas.

RESUME n Pag. 159.

Se utiliza para terminar una rutina de manejo de error; el programa continúa en la línea definida por n. Si se omite el número de línea n o es cero, el control de programa se regresa a la línea en que ocurrió el error; si se indica NEXT el programa continúa en la siguiente línea al error.

```
20  RESUME 100

20  RESUME NEXT
```

RETURN Pag. 153.

Termina una subrutina y regresa el control a la línea siguiente al GOSUB que llamó a la subrutina.

- RUN n Pag. 130.

Causa que sea ejecutado el programa en memoria a partir de la línea n. Si no se especifica el número de línea, el programa se ejecuta desde el principio. Siempre que se ejecuta la instrucción RUN se produce también un CLEAR para evitarlo, se puede utilizar un GOTO.

```
RUN

RUN 100
```

- SET (x,y) Pag. 185

Prende un punto en la pantalla determinado por las coordenadas X horizontal y Y vertical. Las coordenadas x son numeradas de izquierda a derecha del cero al 127. Las coordenadas Y se numeran de arriba a abajo del cero al 47.

```
SET(B+1,20)
```

- STOP Pag. 152.

Detiene la ejecución de un programa e indica el número de línea de la interrupción con la palabra BREAK. El programa se puede continuar con la instrucción CONT.

- SYSTEM Pag. 130.

Pone a la computadora en modo Monitor para cargar programas en lenguaje de máquina que están en cinta.

- TRON Pag. 131.

Enciende la función TRACE de la computadora para depuración y análisis de ejecución de un programa. Prendiendo esta función, al ejecutarse un programa aparece en la pantalla el número de línea que se está ejecutando entre paréntesis.

```
TRON: RUN
```

- TROFF Pag. 131.

Apaga la función TRACE que fué encendida con TRON.

III. FUNCIONES.

Los tipos y rangos de los argumentos usados en las funciones se representan por las siguientes letras:

x:	(-1X10E38,-1X10-38),(1X10E-38,1X10E38)
c:	(0,255)
n:	(-32768, -32767)
str:	argumento string
var:	nombre de variable

Al igual que las proposiciones y declaraciones, las funciones pueden ser utilizadas tanto en modo directo como en modo ejecución. Todas están referidas al manual "TRS-80 Model III Operatión and BASIC Language Reference Manual".

- ABS (x) Pag. 179.

Regresa el valor absoluto del argumento. ABS(x)=X para X mayor o igual a cero, ABS(x)=-X, para X menor que cero.

```
100 IF ABS(x) = 32 PRINT "32"
```

- ASC (str) Pag. 164.

Regresa el código ASCII del primer caracter del string especificado.

El argumento también puede ser una expresión que involucre operadores de strings o funciones.

```
PRINT ASC("A")

100 PRINT ASC(LEFT$(T$,1)
```

- ATN (x) Pag. 179.

Regresa el arco cuya tangente es el argumento, en radianes.

```
100  Y = ATN (B/C)
```


- CDBL (x) Pag. 180.

Regresa una representación de doble precisión del argumento.

```
FOR 1% = 1 TO 25 : PRINT 1/CDBL(1%) : NEXT
```

- CHR$ (c) Pag. 164.

Regresa un caracter cuyo número de código ASCII es el argumento. El argumento puede ser también una expresión aritmética y debe encontrarse en el rango de 0 a 255.

```
100  A$ = CHR$(34)

     PRINT CHR$(193)
```

- CINT (n) Pag. 180.

Regresa el más grande entero no mayor que el argumento. CINT(1.5) = 1, CINT (-1.5) = -2.

```
100 K% = CINT(X#) + CINT(Y#)
```

- COS (x) Pag. 180.

Regresa el coseno del argumento; el argumento debe ser dado en radianes.

```
PRINT COS(THETA + 45 *0.01745329)
```

• CSNG (x) Pag. 180.

Regresa una representación en precisión sencilla del argumento.

```
PRINT CSNG (A# + B#)
```

• ERL Pag. 187.

Regresa el número de la línea en donde fué encontrado un error; se utiliza en rutinas de manejo de error.

```
100 IF ERL = 30 THEN PRINT "ERROR EN LINEA 30"
```

• ERR Pag. 197.

Nos da un valor relativo al número de error encontrado, se utiliza comúnmente en rutinas de manejo de error. El valor que nos da es definido por:

valor regresado = (código de error -1)^2

Por lo que para tener el código real debemos usar ERR/2+1.

```
150 E = ERR/2 + 1
200 IF  E = 12 THEN 600 ELSE 800
```

• EXP (x) Pag. 181.

Regresa el exponencial natural del argumento, es la inversa de logaritmo natural.

```
100 X = EXP(-Y)
```


• FIX (x) Pag. 181.

Regresa el valor del argumento sin la parte decimal que pudiera tener. Fix(2.2) = 2.

```
100 Y = ABS(A - FIX(A))
```

• FRE (número) Pag. 165.

Al igual que MEM, regresa la cantidad disponible de memoria.

```
PRINT FRE(10)
```

• FRE (str) Pag. 165.

Regresa la cantidad de espacio de memoria actualmente disponible para strings.

```
500 PRINT FRE(A$)
```

• INKEY$ Pag. 166.

Nos regresa el último caracter tecleado mediante un chequeo del tablero. Cuando no se teclea algo, la función regresa un string nulo (de longitud cero).

```
100 PRINT "PRESIONA ENTER"
110 A$ = INKEY$
120 IF A$ = CHR$ (13) GOSUB 1000
130 GO TO 110
```

- INP (c) Pag. 188.

Regresa un valor-byte del puerto especificado. Hay 256 puertos, numerados del 0 al 255.

```
PRINT INP(50)
```

Trae un byte del puerto 50 e imprime su valor decimal.

- INT (x) Pag. 181.

Regresa una representación entera del argumento, usando el más grande número completo que no exceda el argumento.

```
100 Z = INT(A*100+0.5)/100
```

- LEFT$(str) Pag. 167.

Regresa el tamaño en caracteres del string especificado.

```
20 PRINT LEN(N$)
```

- LOG (x) Pag. 181.

Regresa el logaritmo natural del argumento. El argumento debe ser positivo.

```
10 LN = LOG (X)

PRINT LOG(2)/LOG(10)
```

Imprime el logaritmo en base 10 de 2.

- MEM Pag. 188.

Regresa el número de bytes de memoria no usados y no protegidos, no incluye el espacio no utilizado de strings.

```
PRINT MEM
```

- MID$(str,posición, longitud) Pag. 168.

Regresa una porción del estring especificado de longitud de la longitud determinada y a partir de la posición dada.

```
PRINT MID$(N$,P,L)
```

- PEEK (n) Pag. 189.

Nos regresa el contenido de la localidad de memoria especificada por n en decimal.

```
V = PEEK(18520)
```

- POINT (x, y) Pag. 186.

Prueba si el punto de la pantalla determinado por las coordenadas x horizontal, y vertical está prendido; si hay un cuadro encendido regresa un -1, si está apagado un 0.

```
IF POINT(50,28) THEN PRINT "ON"
```

- POS (x) Pag. 190.

Regresa un número del 0 al 63, indicando la posición actual del cursor en la pantalla. El argumento x es

```
PRINT TAB(10); POS(0)
```

- RIGHT$ (str,c) Pag. 168.

Regresa los últimos c caracteres del string especificado.

```
PRINT RIGHT$(N$,2)
```

- RND (n) Pag. 182.

Genera un número pseudo random con los rangos siguientes:

Entre 0 y 1 si se utiliza RND(0)

Entre 0 y n si n es mayor que cero.

```
FOR I=1+x20:PRINT RND(1):NEXT
```

- SGN (x) Pag. 182.

Nos da el signo del argumento:

-1 si x es menor que cero

0 si x es igual a cero

1 si x es mayor que cero

- SIN (x) Pag. 183.

Calcula el seno del argumento dado en radianes.

```
SN = SIN(THETA)
```

- SQR (x) Pag. 183.

Calcula la raíz cuadrada del argumento.

```
R1 = SQR(b*B - 4*A*C)
```

- STR$ (x) Pag. 168.

Convierte la expresión numérica dada por el argumento en un string.

```
100  L = LEN(STR$(325.50))
```

- STRING$ (Longitud, caracter o número) Pag. 169.

Regresa un string de longitud determinada, compuesto de caracteres como el especificado. Si se especifica un número, los caracteres serán de' tipo que corresponde al código ASCII.

```
PRINT STRING$(50,"*")

A$ = STRING$(100,42)
```

- TAN (x) Pag. 183.

Cálcula la tangente del argumento dado en radianes.

```
100 X = TAN(2*A)
```

- TIME$ Pag. 170.

Regresa un string con la fecha y la hora actual. Para utilizar el reloj, primero se debe actualizar.

```
PRINT 600, TIME$
```

- USR (x) Pag. 191.

Permite llamar una subrutina en lenguaje de máquina, y después continuar la ejecución del programa BASIC. El argumento puede ser mudo o se puede utilizar para comunicación entre la subrutina y el programa BASIC. Ver páginas 59 a 80 del manual.

- VAL (str) Pag. 170.

Realiza la inversa de la función STR$: regresa el número representado por el string dado.

```
PRINT VAL("100 PESOS")    Imprime 100
```


- VARPTR (va) Pag. 193.

Regresa la dirección de memoria en que se encuentra almacenado el valor de la variable dada.

```
IN = VARPTR(A$)

PRINT VARPTR(M)

X = USR(VARPTR(Y))
```

IV. COMANDOS DE EDICION.

BASIC, incluye un editor para corregir líneas de programa. Para editar una línea se teclea primero el comando.

```
EDIT n.
```

En donde n especifica el número de línea que se desea editar.

Cuando el editor está trabajando en una línea de programa, éste visualiza el número de línea que será editada.

En el modo EDITOR, el teclado acepta caracteres-orientados, esto es, toma caracteres tan pronto como estos son tecleados, sin esperar que se presione la tecla ENTER.

Para mayores detalles, consultar las páginas 195 a 201, del Manual de Referencia (TRS-80 Model III, operation and BASIC Language Reference Manual).

Subcomandos. Para lo siguiente, n es un número entero y c es cualquier caracter.

A Cancela los cambios hechos a una línea y comienza de nuevo la edición; no sale del modo EDICION.

nC Cambia la cantidad de caracteres indicada por n.

nD Borra tantos caracteres como indique n.

E Termina la edición y salva todos los cambios que se hallan hecho.

H Para truncar una línea a partir del lugar donde este el cursor al presionar H; después de esto, se puede insertar otros caracteres.

I Con esto podemos insertar caracteres en el lugar en donde lo indiquemos; esto es, colocando el cursor en el lugar deseado y presionando I.

nKc Elimina todos los caracteres que se encuentran antes de la n-sima ocurrencia del caracter c; el cursor se sitúa en la posición del caracter c. El caracter c no es borrado.

L Despliega la línea completa que se está editando.

Q Con este subcomando salimos del modo EDICION y se cancelan todos los cambios que se hallan hecho a la línea.

nSc Busca la n-sima ocurrencia del caracter c y sitúa al cursor en tal caracter.

X Despliega el escape de un subcomando, esto es, seguimos en modo EDICION pero ya no actúa el último subcomando que usamos.

ENTER Actualiza todos los cambios que hallamos hecho y sale del modo EDICION.

n BARRA ESPACIADORA Mueve el cursor n espacios a la derecha.

n ← Mueve el cursor n espacios a la izquierda.

V. SUBRUTINAS ROM.

La ROM (Memoria solo leible) de la modelo III, contiene muchas subrutinas que pueden ser llamadas por un programa ensamblador Z80 o por un programa BASIC, por medio de la función USR.

Mayor información se encuentra en las páginas 60 a 80 y 191 del manual "TRS-80 Model III,Operatión and BASIC Language Reference Manual"

De acuerdo a sus funciones tenemos (aparece el nombre y la dirección en que se encuentra, tanto en forma decimal como hexadecimal):

Control del sistema.

$CLKON 664/X'0298'

Visualiza un reloj de tiempo real en la esquina superior derecha de la pantalla.

$CLKOFF 673/X'02A1'

Apaga el reloj prendido con $CLKON.

$DATE 12339/X'3033'

Nos da la fecha con que fué actualizado el reloj interno de la máquina.

$DELAY 96/X'0060'

Hace una pausa durante un tiempo especificado.

$INIT:O 105/X'0069'

Inicializa todos los controladores de ENTRADA/SALIDA a sus condiciones normales.

$READY 6681/X'1A:9'

Estando en un programa en lenguaje de maquina, esta rutina regresa a BASIC Modelo III desplegando "ready" en pantalla.

$RESET 0/X'0000'

Inicializa el sistema completo, comenzando con la proposición "Cass?".

$ROUTE 108/X'006C'

Cambia el nombre lógico de dispositivos de ENTRADA/SALIDA.

El uso de esta rutina se puede ver en las páginas 49 a 51.

$SETCAS 12354/X'3042'

Se usa para modificar la velocidad de transferencia de datos (Baud). Al correr aparecerá en pantalla el mensaje:

Cass?

y se debe actuar como al encender la computadora.

$TIME 12342/X'3036'

Nos da la hora actual (para esto, antes se debe actualizar el reloj interno de la computadora).

ENTRADA/SALIDA para Cassette.

$CSHIN 662/X'0236'

Busca y lee el encabezado y el byte de sincronización de una grabación en cinta.

$VDIN 565/X'0235'

Transfiere un byte de la cinta (cassette) a la computadora.

$CSOFF 504/X'01F8'

Apaga la cassettera.

$CSHWR 647/X'0287'

Escribe el encabezado y el byte de sincronización en cinta; para lo cual, primero hay que encender la cassettera.

$CSOUT 612/X'0264'

Transfiere y escribe un byte a la cinta.

ENTRADA DE CARACTERES DEL TECLADO.

$KBCHAR 43/X'002B'

Transfiere un caracter del teclado a memoria, si hay alguno disponible. El caracter no es visualizado.

$KBWAIT 73/X'0049'

Aguarda por un caracter en el teclado. Si se presiona BREAK, este se transfiere como un caracter cualquiera. El caracter teclado no se visualiza.

$KBLINE 64/X'0040'

Espera por una línea completa, terminada con RETURN y la regresa como resultado. Los caracteres tecleados sí se visualizan.

$KBBRK 653/X'028D'

Busca unicamente por la tecla BREAK y la regresa como resultado.

* SALIDA A IMPRESORA.

$PRCHAR 59/X'003B'

Transfiere un caracter a la impresora, si esta no esta disponible espera a que lo esté.

$PRSCN 473/X'01D9'

Esta rutina copia los 1023 caracteres de la pantalla a la impresora, si la impresora no está disponible espera a que lo esté.

* SALIDA A LA PANTALLA DE VIDEO.

$VDCHAR 51/X'0033'

Visualiza un caracter en la posición actual del cursor en la pantalla.

$VDCLS 457/X'01C9'

Limpia completamente la pantalla de video.

$VDLINE 539/X'021B'

Visualiza una línea completa en pantalla. La línea debe terminar con un "retorno de carro" (X'0D') ó ETX(X'03'); el primero es impreso, el segundo no.

* ENTRADA/SALIDA DE LA INTERFASE RS232C.

$RSINIT 90/X'005A'

Inicializa la interfase RS232C.

$RSRCV 80/X'0050'

Recibe un caracter de la interfase RS232C.

$RSTX 85/X'0055'

Transmite un caracter a la interfase RS232C.

VI. DIRECCIONES IMPORTANTES DE RAM.

Colocando varios valores en las direcciones listadas abajo, se pueden activar o controlar muchas de las posibilidades de TRS-80 Modelo III. Para más información de su uso, consulter la función POKE o ver las páginas 189 a 190 del Manual de Referencia.

A continuación se dan direcciones de memoria, tanto decimal como hexadecimal, uso y contenido inicial de ellas.

16409/X'4019' Para alternar mayúsculas y minúsculas. Colocando un 0 tenemos minúsculas y mayúsculas; contenido diferente de 0 permite solo mayúsculas. Contenido inicial "Mayúsculas".

16412/X'401C' Control del cursor. Con un 0 el cursor es intermitente diferente de cero el cursor permanece fijo. Contenido inicial intermitente.

16416/X'4020' Dirección del cursor. Para situar al cursor en determinada posición.

16419/X'4023' Código del caracter ASCII que representa al cursor. Inicialmente es el 176.

16424/X'4028' Número máximo de líneas por página más uno en la impresora. Inicialmente 67.

16425/X'4029' Número de líneas impresas más uno. Inicialmente 1.

16427/X'4028' Máxima longitud de línea en la impresora menos dos. El número máximo es 255.

16526/X'408E' Dirección de la rutina USR, abarca dos bytes: 16526 y 16527. Inicialmente contiene el 7754.

16913/X'4211' Para seleccionar velocidad de transferencia de datos; teniendo 0 tenemos 500 Baud, diferente de cero 1500 Baud.

16916/X'4214' Protección de las 7 líneas superiores de la pantalla; puede tener de 0 al 7 y valores mayores se interpretan como módulo 8.

16919/X'4217' Contiene la fecha y la hora en seis bytes. Del byte 16919 al 16924, se tienen respectivamente segundos, minutos, hora, año, día y mes.

16928/X'4220' Se utiliza en la subrutina $ROUTE y contiene el dispositivo destino en dos bytes, 16928 y 16929.

16930/X'4222' Se utiliza en la subrutina $ROUTE y contiene el dispositivo fuente en dos bytes, 16930 y 16931.

VII. CODIGOS DE CONTROL DE VIDEO.

Consultar la página 228 del Manual de Referencia. A continuación se dan los códigos que tienen efecto en TRS-80 Modelo III. El código se da en decimal y hexadecimal.

8/X'08' Regresa el cursor y borra el caracter.

9/X'09' Avanza el cursor al siguiente campo de tabulación.

10/X'0A' Mueve el cursor al inicio de la siguiente línea, borra la línea y provoca un "retorno de carro".

13/X'0D' Actúa igual que 10/X'0A'

14/X'0E' Prende el cursor. En impresora se utiliza para imprimir caracteres de doble ancho.

15/X'0F' Apaga el cursor.

21/X'15' Alterna los caracteres especiales/compresión.

22/X'16' Alterna los caracteres especiales.

23/X'17' Cambia a caracteres de doble ancho.

24/X'18' Regresa el cursor sin borrar.

25/X'19' Avanza el cursor.

26/X'1A' Baja el cursor.

27/X'1B' Sube el cursor. Al usar la impresora equivale a ESCAPE.

28/X'1C' Coloca al cursor en la esquina superior izquierda.

29/X'1D' Regresa el cursor al principio de la línea y borra la misma.

30/X'1E' Borra hasta el fin de la línea.

31/X'1F' Borra hasta el fin de la pantalla.

VIII. MENSAJES DE ERROR. Código.

Ya sea para identificar un error al correr un programa o para fabricar rutinas de manejo de error, se dan los siguientes códigos, abreviaciones y significados. Consultar el Manual de Referencia, páginas 223 a 225, para más información.

CODIGO	ABREVIATURA	SIGNIFICADO
1	NF	Se encontró un NEXT sin haberse registrado un FOR.
2	SN	Error de sintaxis.
3	RG	Se encontró un RETURN sin registrarse antes un GOSUB.
4	OD	No existen datos para ejecutar un READ o un INPUT#.
5	FC	Se intenta usar una función utilizando un parámetro ilegal.
6	OV	Overflow. La magnitud de un número leído o derivado es más grande de lo que la computadora puede manejar..
7	OM	Toda la memoria disponible ha sido usada o reservada.
8	UL	Se intenta referir o brincar a una línea que no existe.

CODIGO	ABREVIATURA	SIGNIFICADO
9	BS	Se intenta asignar o hacer referencia a un elemento con un índice más grande que el dimensionado con la instrucción DIM.
10	DD	Se intenta dimensionar una variable que anteriormente fué ya dimensionada.
11	/0	Se intenta hacer una división con un divisor igual a cero.
12	ID	Uso ilegal de un comando, como INPUT en modo directo.
13	TIM	Se intenta hacer una asignación con operadores incompatibles, esto, es un string con un no string y viceversa.
14	OS	Fuera del espacio localizado para una variable de tipo "string".

CODIGO	ABREVIATURA	SIGNIFICADO
15	LS	Se intente asignar a una variable string un valor que excede de 255 caracteres.
16	ST	Una operación con "strings" fué demasiado compleja. Divídase en pasos más cortos.
17	CN	El programa no puede continuar.
18	NR	No se puede ejecutar un RESUME.
19	RW	Se encontro una proposición RESUME sin registrarce antes un ON ERROR GOTO.
20	UE	Se inentó generar un error con un código ilegal.
21	MO	Falta un operando al ejecutar una instrucción.
22	FD	Existe error en la entrada de un archivo de datos de una fuente externa (cassettera).

CODIGO	ABREVIATURA	SIGNIFICADO
23	L3	Se trata de utilizar una instrucción solo permisible en sistema de DISCO.

IX. CARACTERES ESPECIALES.

'	Abreviatura de REM
%	Hace variables de precisión entera.
!	Hace variables de precisión simple.
#	Hace variables de precisión doble.
$	Hace variables tipo "string".
:	Separa instrucciones en una misma línea.
?	Igual que PRINT (no es válido L? por LPRINT)
,	Para escribir cada espacio de tabulación.
;	Para escribir sin dejar espacio.

X. OPERADORES.

En orden de procedencia:

↑ ó [	Exponenciación
- , +	Signo unario negativo y positivo.

* , /	Multiplicación, división.
+ , -	Suma y concatenación, resta.
<,>,=, <=,>=,<>	Operadores relacionales.
NOT	
AND	
OR	
—	

TRS - 80 MODEL III
MICROCOMPUTER SYSTEM

GUIA DE REFERENCIA

CONTENIDO.

TRS - 80 MODEL III
MICROCOMPUTER SYSTEM

GUIA DE REFERENCIA

I. ENCENDIDO.

Al iniciar, todos los periféricos y la computadora deberán estar apagados.

1. Primero encender los periféricos (impresora normalmente) y después encender la computadora.

2. Deberá aparecer el mensaje.

   CASS?

   Para seleccionar alta velocidad de cassette (1500 Bits/seg), se presiona H o ENTER. Normalmente se usa esta velocidad.

   Para una velocidad de transmisión baja (500 Bits/seg) se debe presionar L , velocidad usada para salvar o cargar programas BASIC nivel II.

3. Después aparece el mensaje

   MEMORY SIZE?

   Esto nos permite reservar memoria, para utilizar la totalidad de ella sólo se presiona ENTER.

Para reservar memoria, se teclea en decimal la mayor dirección de memoria que se desee usar y se termina con ENTER.

4. Finalmente deberá aparecer el mensaje

```
MODEL III BASIC

(c) TANDY '80

READY
```

Y la computadora estará lista para su uso.

II. PROPOSICIONES Y DECLARACIONES.

Las siguientes proposiciones pueden ser utilizadas tanto en modo directo (sin etiqueta) como en modo ejecución (con etiqueta y por medio de la instrucción RUN).

Todas las proposiciones se encuentran referidas a la página en que se encuentran en el manual TRS-80 MODEL III operation and BASIC Language Reference Manual.

AUTO inicio, incremento Pag. 125

Nos da una secuencia automática de líneas numeradas.

AUTO inicia en 10 y se incrementa en 10

AUTO5,5 inicia en 5 y se incrementa en 5

AUTO,5 inicia en 0 y se incrementa en 5

AUTO50, inicia en 50 y se incrementa en 10

- CLEAR n Pag. 126

Inicializa todas las variables a cero (únicamente CLEAR). Con argumento reserva n bytes a todas las variables de tipo string.

CLEAR 100 reserva 100 bytes.

Se debe tener en cuenta que al ejecutar esta instrucción las variables declaradas con DFNSTR regresan a su tipo original.

- CLOAR "nombre" Pags. 29-34, 126.

Carga a la computadora un programa BASIC que se encuentra en cinta; al utilizarlo deberá estar conectada a la computadora la casettera con el cassette en que se halla el programa montado por el lado debido, y con la tecla "PLAY" oprimida.

Sólo es usado el primer caracter del nombre.

CLOAD Carga el primer programa que se encuentre.

CLOAD"PRO" Carga el primer programa que halla sido grabado con un nombre iniciado con P.

CLOAD?"P" Compara el programa "P" en cinta con el programa que en ese momento reside en la computadora, byte por byte.

Nos sirve para verificar una correcta transferencia después de salvar un programa en cinta. Pag. 127.

- CLS Pag. 186.

Borra todo lo que hay en la pantalla y sitúa al cursor en la esquina superior izquierda.

- CONT Pag. 127.

Continúa la ejecución de un programa que ha sido interrumpido con la instrucción STOP o la tecla BREAK.

La ejecución continúa a partir del estado que se tenía al suspender el programa, a menos que se haya modificado alguna variable en modo de ejecución directa.

- CSVE "nombre" Pag. 29-34, 127.

Guarda un programa residente en cinta. El nombre puede formarse por letras, números o caracteres especiales (excepto comillas), sin embargo, solo se tomará encuenta el primer caracter encontrado.

CSAVE "I/HOLA" Guarda el programa residente en cinta, con el nombre " I "

- DATA Pag. 142.

Almacena datos que serán accesados por medio de una proposición READ. Los elementos del DATA serán leídos secuencialmente a partir de la primera proposición data. Los datos de tipo string no necesitan encerrarse entre comillas a menos que contengan blancos o comas.

```
DATA UN DATO, "Y/O ESTE",123
```

- DEFDBL variables, rango Pag. 149.

Define variables de doble precisión que permite 17 dígitos de precisión. Si se define un rango, vgr. A-F, todas las variables que comiencen con una letra de ese rango serán de doble precisión.

```
DEFDBL A-F, X
```

- DEFINT variables, rango Pag. 148.

Define variables de tipo entero. Ayuda a ahorrar memoria, puesto que los valores enteros ocupan menos espacio de ella. Todas las variables que comiencen con una letra definida entera, serán de tipo entero.

```
DEFINT I-N
```

- DEFSNG variable, rango Pag. 148.

Define variables de tipo simple precisión, esto es, solo se guardarán 7 dígitos en memoria. Funciona como DEFDBL y DEFINT, y al igual que en ellas se debe notar que aunque alguna letra este definida con ese tipo, éste puede ser cambiado con los caracteres de declaración de tipo.

DEFSNG A-F — Define el rango A a F como simple precisión.

Sin embargo ALFA# será tratada como variable de doble precisión.

- DEFSTR variable, rango Pag. 148.

Define variables de tipo "string". Normalmente las variables de tipo "string" pueden contener hasta 50 caracteres; esto se puede cambiar con la proposición CLEAR n.

DEFSTR W,D-F

- DELETE número de secuencia o rango Pag. 128.

Elimina de memoria las líneas de programa especificadas. Se puede referir a una línea o a una secuencia de líneas.

DELETE 80 — Borra de memoria la línea 80

DELETE-80 Borra desde el principio del programa hasta la línea 80 inclusive.

DELETE 10-80 Borra de la línea 10 a la 80.

DELETE Borra la última línea editada o grabada en memoria.

- DIM Var (dimensión) Pag. 150-151,173-178.

Dimensiona uno o más arreglos; la dimensión de cada arreglo será de 0 al número que se especifique.

DIM A,B(3),D(2,2) Reserva memoria para los arreglos A y B que tendrán de 0 a 3 suscriptores, esto es, A(0), A(1), A(2), A(3). La variable D será una matriz de 3 x 3 elementos.

Cuando se omite la dimensión de alguna variable, ésta tiene 11 elementos (0 a 10).

También se pueden dimensionar variables de tipo "string".

Si en algún lugar de un programa se redimensiona una variable, primero se debera ejecutar la instrucción CLEAR, de lo contrario se producira un error por dimensionar dos veces la misma variable.

- EDIT número de línea Pag. 14,128,195-201.

Cambia la computadora a modo de edición. En este modo se puede corregir o alterar líneas del programa residente. Consultar comandos de edición.

EDIT 100 La línea 100 aparecerá para ser editada.

EDIT La última línea referenciada aparece para ser editada.

- END Pag. 151.

Termina normalmente la ejecución de un programa, se utiliza principalmente para provocar que la ejecución termine en un lugar del programa que no es precisamente la última línea de él.

- ERROR(n) Pag. 158.

Simula el error referido con el número n como parámetro. Su principal uso es en la prueba de rutinas de error (vease ON ERROR GOTO), la computadora procede exactamente como si ese error hubiera ocurrido (consultar la tabla de errores. (Pag. 223-225).

ERROR(2) Simula un error de sintaxis.

- FOR - TO - STEP - ... NEXT Pag. 155-157.

Provoca la iteración de las instrucciones que se encuentran entre FOR-TO-STEP y NEXT tantas veces como lo permita el límite. Al ejecutarce una instrucción de este tipo primero se realiza lo que esté antes del NEXT, al llegar ahí se incrementa el contador y se compara con el límite; en caso de ser mayor a él, continúa con la siguiente instrucción, de lo contrario se regresa a la primera después de FOR. Si se omite el incremento "STEP", se toma como 1. Es posible utilizar incrementos fraccionales y negativos.

```
FOR I=1 TO 10 STEP 0.5
    PRINT I;
NEXT I
```

- GOSUB número de línea Pag. 153.

Transfiere el control del programa a la línea especificada a partir de dicha línea en adelante, se considerará como una subrutina hasta encontrar la proposición RETURN que regresará el mando a la siguiente instrucción después del GOSUB.

```
GOSUB 1000      Transfiere el control a la línea
                1000.
```

- GOTO número de línea , Pag. 152.

Transfiere el control del programa a la línea especificada. El programa continúa su ejecución normal; en modo inmediato, se puede usar para iniciar la corrida de un programa en una línea determinada.

GOTO 1000 Transfiere el control a la línea 1000.

- IF - THEN - ELSE . Pag. 160-161.

Prueba una expresión lógica o relacional y:

Si es verdadera ejecuta la instrucción o instrucciones que se encuentran entre THEN y ELSE.

Si es falsa ejecuta la ó las instrucciones que están después de ELSE.

```
IF A$="SI" THEN PRINT"CONTINUA":PRINT"ARRANCA"
ELSE PRINT"FINALIZA":END
```

En el caso de que A$ sea igual a la cadena "SI", se imprimirá CONTINUA y ARRANCA, de lo contrario imprimirá FINALIZA y dará por terminado el programa.

Al utilizar el ELSE se debe cuidar continuar en la misma línea lógica (sin oprimir ENTER).

Es posible omitir la opción ELSE. En este caso, al ser verdadera la expresión evaluada se ejecuta la proposición indicada (en caso de tener varias instrucciones separadas por dos puntos en una misma línea se ejecutan todas ellas) y cuando la expresión es falsa, se produce un brinco a la siguiente línea lógica.

IF P=Q THEN 200 — De ser verdadera se irá a la línea 200.

- INPUT Lista de variables — Pag. 140-141.

Causa la suspensión del programa hasta que se hayan dado por medio del teclado los valores de las variables especificadas en la lista. La lista de variables puede incluir valores numéricos y cuerdas. Al teclear los datos, estos deberán ir separados por comas, y en caso de cuerdas que contengan blancos, comas o puntos, se deberán encerrar entre comillas.

INPUT A,A$ — Pedirá el valor numérico de A y la cuerda A$.

Como ayuda, es posible desplegar un mensaje en la pantalla al solicitar un INPUT.

INPUT"DAME TU NOMBRE"; NO$

- INPUT #-1 lista de variables Pag. 145.

Se utiliza para leer valores guardados en un cassette. Los valores deben coincidir en cantidad, orden y tipo con los que son solicitados en el INPUT. Se debe situar la cinta al inicio de los valores a leer y tener la grabadora con el "PLAY" encendido.

- LET asignación. Pag. 151.

Se utilizaba en BASIC standard; no es necesario para esta máquina.

```
LET A = B*C
```

- LIST secuencia. Pag. 13, 31, 35, 128.

Causa la visualización en pantalla de la línea o líneas que se le indique.

LIST 50	Aparece la línea 50
LIST 50-100	Aparece de la línea 50 a la 100
LIST 50-	Aparece de la línea 50 al fin del programa.
LIST.	Aparece la última línea referida.

- LLIST secuencia Pag. 129.

Actúa como LIST, solo que la visualización se hace en la impresora de papel.

- LPRINT lista de variables. Pag. 144.

Actúa como PRINT, solo que la impresión es a papel.

LPRINT TAB(t) Pag. 144.

Funciona como PRINT TAB, solo que la impresión es a papel.

- LPRINT USING

Similar a PRINT USING, impresión a papel.

- NEW Pag. 129.

Borra el programa residente, inicializa todas las variables a cero (nulo). Se utiliza para iniciar un nuevo programa.

- ON ERROR GOTO número de línea Pag. 158.

Cuando la computadora encuentra un error suspende el programa, a menos de que una instrucción ON ERROR haya estado antes de la línea de error; en ese caso la secuencia de programa brincará a la línea que se le indique (normalmente es una rutina de manejo de errores).

ON ERROR GOTO 1000

Para deshabilitar una rutina de manejo de error, se utiliza lo siguiente: ON ERROR GOTO Ø

ON - GOSUB lista de etiquetas Pag. 155.

Es un salto condicional a rutinas del programa. Evalúa una variable o expresión y dependiendo de su valor transfiere el control al número de línea que se encuentra en el lugar correspondiente dentro de la lista.

El resultado de la evaluación se debe encontrar entre 0 y 255, de lo contrario produce error. Si el resultado de la evaluación tiene parte fraccionaria, sólo se tomará la parte entera.

ON N GOSUB 100, 200, 300

Si N = 0 continúa normalmente

N = 1 va a la línea 100

N = 2 va a la línea 200

N = 3 va a la línea 300

Proseguirá a partir de la línea a donde se hizo el salto hasta encontrar la proposición RETURN que regresa el control a la siguiente instrucción al ON GOSUB.

- ON - GOTO lista de etiquetas

Salto condicional a líneas simples de programa. Funciona similar al ON-GOSUB, solo que no se puede regresar el control usando RETURN.

```
ON SNG(X) + 2 GOTO 200, 300, 400
```

- OUT punto, valor Pag. 189.

Envía el valor-byte al puerto especificado. Puerto y valor están en el rango de 0 a 255.

- POKE n, v Pag. 189.

Guarda el valor v (entre 0 y 255) en la dirección de memoria n (entre 0 y 32767). De este modo podemos, por ejemplo; hacer que el cursor no "parpadee". Usamos

```
POKE 16412,1
```

- PRINT lista Pag. 133.

Sirve para visualizar información en la pantalla. La lista puede ser mensajes encerrados entre comillas, variables string, números constantes, variables o expresiones que contengan cualquiera de los elementos anteriores. Los elementos de la lista pueden estar separados por comas o punto y coma; la coma produce un avance a la siguiente zona de impresión.

Si se utiliza punto y coma, se inserta un espacio después de escribir un valor numérico, después de escribir un string no inserta espacios. El punto y como al final de una lista de PRINT suprime el regreso automático del cursor.

La pantalla se divide en cuatro zonas de impresión, teniendo cada una de ellas dieciseis espacios.

```
PRINT "LA RESPUESTA ES:";R$$

PRINT A,B,C;
```

La instrucción PRINT puede tener los siguientes modificadores:

n — Comienza una impresión en la posición n de la pantalla; n debe ser un número entre 0 y 1023. /pag.134/

```
PRINT  509,"CENTRO"
```

TAB(n) — Mueve el cursor a una posición especificada por n. La posición n puede estar dada por una expresión y será hasta 127. Es posible utilizar más de un TAB en una instrucción PRINT. /pag. 135/.

```
PRINT TAB(32)   "HOLA"
PRINT TAB(2*X)  "HOLA;TAB(2*X+5)N$
```

USING — Permite especificar el formato en que se hará una impresión. Tiene la siguiente forma:

PRINT USING string; valor(es)

En donde string es una cadena con especificadores de campo que define la forma en que se imprimirá lo que precede al punto y coma. /pag. 136 - 140/.

ESPECIFICADORES DE CAMPO COMO COMPLEMENTO DE USING

\# — Para dar formato a números. El número de símbolos # usados especifica el campo que tendrá el número. Si el campo es mayor que el número, las posiciones no usadas del campo se imprimen como espacios a la izquierda, a menos que este después del punto decimal en cuyo caso se imprimirán ceros a la derecha. Cuando el número excede el tamaño del campo, aparecerá completo y precedido por el símbolo %.

. — Indica la posición del punto decimal.

,	Colocada en cualquier posición entre el primer dígito y el punto, causa que se imprima una coma a la izquierda de cada 3 dígitos.
**	Colocados al principio del campo, provoca que todos los espacios no usados a la izquierda sean llenados por asteriscos.
$	Se imprime un signo de pesos en la primer posición del campo.
$$	Si se colocan dos signos de pesos, el signo de pesos se imprime a la izquierda del primer dígito.
**$	Causa que todas las posiciones no usadas sean llenadas con asteriscos y el signo de pesos se imprime junto al primer dígito.
^^^^	Causan que el número sea impreso en forma exponencial (E o D).
+	Cuando se coloca al principio del campo, se imprimirá + para números positivos y para números negativos.

- Colocado al final del campo, produce que se imprima en ese lugar un signo - para números negativos, y un espacio en blanco para números positivos.

% % Especifica un campo para variables o constantes de tipo string y de más de un caracter. La longitud del campo será igual al número de espacios entre los signos % más dos.

! Da campo para el primer caracter del string que se imprima en ese campo.

Cualquier otro caracter que se utilice, simplemente se imprimirá como tal en la posición que ocupe.

Ejemplos:

```
10 A$="###.#"
20 PRINT USING A$;X

   PRINT USING "**$##.#";COSTO

   PRINT USING "##.#[ [ [ [ ";NUM

   PRINT USING "!.!%      %";N1$,N2$,N3$
```

• RANDOM Pag. 182

Da una nueva semilla al generador de números pseudo aleatorios. Se utiliza al principio de un programa cuando deseamos asegurar una secuencia impredecible de nûmeros RANDOM.

READ lista Pag. 142

Asigna a los elementos de la lista de variables los valores definidos en una instrucción DATA. La primer instrucción READ usada en el programa asignará los primeros valores de la primer instrucción DATA y los READ posteriores asignarán los siguientes valores encontrados y así sucesivamente.

```
20  READ A,B,N$
```

• REM Pag. 160.

Se utiliza para introducir comentarios. Utilizando REM, los siguientes 255 caracteres que se escriban, serán ignorados. Se puede utilizar un apóstrofo para abreviar.

```
10  REM *** INICIO ***

20  X = X[ 2 REM  X CUADRADA

30  Y = X[ 3  ' X CUBICA
```

- RESET(x,y) Pag. 186.

Apaga un punto de la pantalla situado por las coordenadas x,y. Ver también SET(x,y).

```
RESET (P,10)
```

- RESTORE Pag. 143.

Regresa el apuntador de los elementos de DATA al primer valor del primer DATA en el programa. Se utiliza cuando quieren usar las mismas líneas de DATA ya utilizadas.

- RESUME n Pag. 159.

Se utiliza para terminar una rutina de manejo de error; el programa continúa en la línea definida por n. Si se omite el número de línea n o es cero, el control de programa se regresa a la línea en que ocurrió el error; si se indica NEXT el programa continúa en la siguiente línea al error.

```
20  RESUME 100

20  RESUME NEXT
```

- RETURN Pag. 153.

Termina una subrutina y regresa el control a la línea siguiente al GOSUB que llamó a la subrutina.

- RUN n Pag. 130.

Causa que sea ejecutado el programa en memoria a partir de la línea n. Si no se especifica el número de línea, el programa se ejecuta desde el principio. Siempre que se ejecuta la instrucción RUN se produce también un CLEAR para evitarlo, se puede utilizar un GOTO.

```
RUN

RUN 100
```

- SET (x,y) Pag. 185

Prende un punto en la pantalla determinado por las coordenadas X horizontal y Y vertical. Las coordenadas x son numeradas de izquierda a derecha del cero al 127. Las coordenadas Y se numeran de arriba a abajo del cero al 47.

```
SET(B+1,20)
```

- STOP Pag. 152.

Detiene la ejecución de un programa e indica el número de línea de la interrupción con la palabra BREAK. El programa se puede continuar con la instrucción CONT.

- SYSTEM Pag. 130.

Pone a la computadora en modo Monitor para cargar programas en lenguaje de máquina que están en cinta.

- TRON Pag. 131.

Enciende la función TRACE de la computadora para depuración y análisis de ejecución de un programa. Prendiendo esta función, al ejecutarse un programa aparece en la pantalla el número de línea que se está ejecutando entre picoparéntesis.

```
TRON: RUN
```

- TROFF Pag. 131.

Apaga la función TRACE que fué encendida con TRON.

III. FUNCIONES.

Los tipos y rangos de los argumentos usados en las funciones se representan por las siguientes letras:

x:	(-1X10E38,-1X103-38),(1X10E-38,1X10E38)
c:	(0,255)
n:	(-32768, -32767)
str:	argumento string
var:	nombre de variable

Al igual que las proposiciones y declaraciones, las funciones pueden ser utilizadas tanto en modo directo como en modo ejecución. Todas están referidas al manual "TRS-80 Model III Operatión and BASIC Languaje Reference Manual".

- ABS (x) Pag. 179.

Regresa el valor absoluto del argumento. ABS(x)=X para X mayor o igual a cero, ABS(x)=-X, para X menor que cero.

```
100 IF ABS(x) = 32 PRINT  "32"
```

- ASC (str) Pag. 164.

Regresa el código ASCII del primer caracter del string especificado.

El argumento también puede ser una expresión que involucre operadores de strings o funciones.

```
PRINT ASC("A")
```

```
100 PRINT ASC(LEFT$(T$,1)
```

- ATN (x) Pag. 179.

Regresa el arco cuya tangente es el argumento, en radianes.

```
100  Y = ATN (B/C)
```

- CDBL (x) Pag. 180.

Regresa una representación de doble precisión del argumento.

```
FOR I% = 1 TO 25 : PRINT 1/CDBL(I%) : NEXT
```

- CHR$ (c) Pag. 164.

Regresa un caracter cuyo número de código ASCII es el argumento. El argumento puede ser también una expresión aritmética y debe encontrarse en el rango de 0 a 255.

```
100  A$ = CHRS$(34)

     PRINT CHR$(193)
```

- CINT (n) Pag. 180.

Regresa el más grande entero no mayor que el argumento. CINT(1.5) = 1, CINT (-1.5) = -2.

```
100 K% = CINT(X#) + CINT(Y#)
```

- COS (x) Pag. 180.

Regresa el coseno del argumento; el argumento debe ser dado en radianes.

```
PRINT COS(THETA + 45 *0.01745329)
```

- CSNG (x) Pag. 180.

Regresa una representación en precisión sencilla del argumento.

```
PRINT CSNG (A# + B#)
```

- ERL Pag. 187.

Regresa el número de la línea en donde fué encontrado un error; se utiliza en rutinas de manejo de error.

```
100 IF ERL = 30 THEN PRINT "ERROR EN LINEA 30"
```

- ERR Pag. 197.

Nos da un valor relativo al número de error encontrado, se utiliza comúnmente en rutinas de manejo de error. El valor que nos da es definido por:

valor regresado = (código de error -1)*2

Por lo que para tener el código real debemos usar ERR/2+1.

```
150 E = ERR/2 + 1

200 IF  E = 12 THEN 600 ELSE 800
```

- EXP (x) Pag. 181.

Regresa el exponencial natural del argumento, es la inversa de logaritmo natural.

```
100 X = EXP(-Y)
```

- FIX (x) Pag. 181.

Regresa el valor del argumento sin la parte decimal que pudiera tener. FIX(2.2) = 2.

```
100 Y = ABS(A - FIX(A))
```

- FRE (número) Pag. 165.

Al igual que MEM, regresa la cantidad disponible de memoria.

```
PRINT FRE(10)
```

- FRE (str) Pag. 165.

Regresa la cantidad de espacio de memoria actualmente disponible para strings.

```
500 PRINT FRE(A$)
```

- INKEY$ Pag. 166.

Nos regresa el último caracter tecleado mediante un chequeo del tablero. Cuando no se teclea algo, la función regresa un string nulo (de longitud cero).

```
100 PRINT "PRESIONA ENTER"
110 A$ = INKEY$
120 IF A$ = CHR$ (13) GOSUB 1000
130 GO TO 110
```

- INP (c) Pag. 188.

Regresa un valor-byte del puerto especificado. Hay 256 puertos, numerados del 0 al 255.

```
PRINT INP(50)        Trae un byte del puerto 50
                     e imprime su valor decimal.
```

- INT (x) Pag. 181.

Regresa una representación entera del argumento, usando el más grande número completo que no exceda al argumento.

```
100 Z = INT(A*100 + 0.5)/100
```

- LEFT$(str) Pag. 167.

Regresa el tamaño en caracteres del string especificado.

```
20 PRINT LEN(N$)
```

- LOG (x) Pag. 181.

Regresa el logaritmo natural del argumento. El argumento debe ser positivo.

```
10 LN = LOG (X)

PRINT LOG(2)/LOG(10)     imprime el logaritmo en
                         base 10 de 2.
```

- MEM Pag. 188.

Regresa el número de bytes de memoria no usados y no protegidos, no incluye el espacio no utilizado de strings.

```
PRINT  MEM
```

- MID$(str,posición, longitud) Pag. 168.

Regresa una porción del estring especificado de longitud de la longitud determinada y a partir de la posición dada.

```
PRINT MID$(N$,P,L)
```

- PEEK (n) Pag. 189.

Nos regresa el contenido de la localidad de memoria especificada por n en decimal.

```
V = PEEK(18520)
```

- POINT (x, y) Pag. 186.

Prueba si el punto de la pantalla determinado por las coordenadas x horizontal, y vertical está prendido; si hay un cuadro encendido regresa un -1, si está apagado un 0.

```
IF POINT(50,28) THEN PRINT "ON"
```

- POS (x) Pag. 190.

Regresa un número del 0 al 63, indicando la posición actual del cursor en la pantalla. El argumento x es

```
PRINT TAB(10); POS(0)
```

- RIGHT$ (str,c) Pag. 168.

Regresa los últimos c caracteres del string especificado.

```
PRINT RIGHT$(N$,2)
```

- RND (n) Pag. 182.

Genera un número pseudo random con los rangos siguientes:

Entre 0 y 1 si se utiliza RND(0)

Entre 0 y n si n es mayor que cero.

```
FOR I=1+x20:PRINT RND(1):NEXT
```

- SGN (x) Pag. 182.

Nos da el signo del argumento:

-1 si x es menor que cero

0 si x es igual a cero

1 si x es mayor que cero

- SIN (x) Pag. 183.

Calcula el seno del argumento dado en radianes.

```
SN = SIN(THETA)
```

- SQR (x) Pag. 183.

Calcula la raíz cuadrada del argumento.

```
R1 = SQR(B*B - 4*A*C)
```

- STR$ (x) Pag. 168.

Convierte la expresión numérica dada por el argumento en un string.

```
100  L = LEN(STR$(325.50))
```

- STRING$ (Longitud, caracter o número) Pag. 169.

Regresa un string de longitud determinada, compuesto de caracteres como el especificado. Si se especifica un número, los caracteres serán del tipo que corresponde al código ASCII.

```
PRINT STRING$(50,"*")

A$ = STRING$(100,42)
```

- TAN (x) Pag. 183.

Cálcula la tangente del argumento dado en radianes.

```
100 X = TAN(2*A)
```

- TIME$ Pag. 170.

Regresa un string con la fecha y la hora actual. Para utilizar el reloj, primero se debe actualizar.

```
PRINT 600, TIME$
```

- USR (x) Pag. 191.

Permite llamar una subrutina en lenguaje de máquina, y después continuar la ejecución del programa BASIC. El argumento puede ser mudo o se puede utilizar para comunicación entre la subrutina y el programa BASIC. Ver páginas 59 a 80 del manual.

- VAL (str) Pag. 170.

Realiza la inversa de la función STR$: regresa el número representado por el string dado.

```
PRINT VAL("100 PESOS")    imprime 100
```

VARPTR (va) Pag. 193.

Regresa la dirección de memoria en que se encuentra almacenado el valor de la variable dada.

```
IN = VARPTR(A$)

PRINT VARPTR(M)

X = USR(VARPTR(Y))
```

IV. COMANDOS DE EDICION.

BASIC, incluye un editor para corregir líneas de programa. Para editar una línea se teclea primero el comando.

```
EDIT n
```

En donde n especifica el número de línea que se desea editar.

Cuando el editor está trabajando en una línea de programa, éste visualiza el número de línea que será editada.

En el modo EDITOR, el teclado acepta caracteres-orientados, esto es, toma caracteres tan pronto como estos son tecleados, sin esperar que se presione la tecla ENTER.

Para mayores detalles, consultar las páginas 195 a 201, del Manual de Referencia (TRS-80 Model III, operatión and BASIC Language Reference Manual).

Subcomandos. Para lo siguiente, n es un número entero y c es cualquier caracter.

A — Cancela los cambios hechos a una línea y comienza de nuevo la edición; no sale del modo EDICION.

nC — Cambia la cantidad de caracteres indicada por n.

nD — Borra tantos caracteres como indique n.

E — Termina la edición y salva todos los cambios que se hallan hecho.

H — Para truncar una línea a partir del lugar donde este el cursor al presionar H; después de esto, se puede insertar otros caracteres.

I — Con esto podemos insertar caracteres en el lugar en donde lo indiquemos; esto es, colocando el cursor en el lugar deseado y presionando I.

nKc Elimina todos los caracteres que se encuentran antes de la n-sima ocurrencia del caracter c; el cursor se sitúa en la posición del caracter c. El caracter c no es borrado.

L Despliega la línea completa que se está editando.

Q Con este subcomando salimos del modo EDICION y se cancelan todos los cambios que se hallan hecho a la línea.

nSc Busca la n-sima ocurrencia del caracter c y sitúa al cursor en tal caracter.

X Despliega el escape de un subcomando, esto es, seguimos en modo EDICION pero ya no actúa el último subcomando que usamos.

ENTER Actualiza todos los cambios que hallamos hecho y sale del modo EDICION.

n BARRA ESPACIADORA Mueve el cursor n espacios a la derecha.

n ← Mueve el cursor n espacios a la izquierda.

V. SUBRUTINAS ROM.

La ROM (Memoria solo leible) de la modelo III, contiene muchas subrutinas que pueden ser llamadas por un programa ensamblador Z80 o por un programa BASIC, por medio de la función USR.

Mayor información se encuentra en las páginas 60 a 80 y 191 del manual "TRS-80 Model III,Operatión and BASIC Language Reference Manual"

De acuerdo a sus funciones tenemos (aparece el nombre y la dirección en que se encuentra, tanto en forma decimal como hexadecimal):

Control del sistema.

$CLKON 664/X'0298'

Visualiza un reloj de tiempo real en la esquina superior derecha de la pantalla.

$CLKOFF 673/X'02A1'

Apaga el reloj prendido con $CLKON.

$DATE 12339/X'3033'

Nos da la fecha con que fué actualizado el reloj interno de la máquina.

$DELAY 96/X'0060'

Hace una pausa durante un tiempo especificado.

$INITIO 105/X'0069'

Inicializa todos los controladores de ENTRADA/SALIDA a sus condiciones normales.

$READY 6681/X'1A19'

Estando en un programa en lenguaje de maquina, esta rutina regresa a BASIC Modelo III desplegando "ready" en pantalla.

$RESET 0/X'0000'

Inicializa el sistema completo, comenzando con la proposición "Cass?".

$ROUTE 108/X'006C'

Cambia el nombre lógico de dispositivos de ENTRADA/SALIDA

El uso de esta rutina se puede ver en las páginas 49 a 51.

$SETCAS 12354/X'3042'

Se usa para modificar la velocidad de transferencia de datos (Baud). Al correr aparecerá en pantalla el mensaje:

Cass?

y se debe actuar como al encender la computadora.

$TIME 12342/X'3036'

Nos da la hora actual (para esto, antes se debe actualizar el reloj interno de la computadora).

- ENTRADA/SALIDA para Cassette.

$CSHIN 662/X'0296'

Busca y lee el encabezado y el byte de sincronización de una grabación en cinta.

$VDIN 565/X'0235'

Transfiere un byte de la cinta (cassette) a la computadora.

$CSOFF 504/X'01F8'

Apaga la cassettera.

$CSHWR 647/X'0287'

Escribe el encabezado y el byte de sincronización en cinta; para lo cual, primero hay que encender la cassettera.

$CSOUT 612/X'0264'

Transfiere y escribe un byte a la cinta.

- ENTRADA DE CARACTERES DEL TECLADO.

$KBCHAR 43/X'002B'

Transfiere un caracter del teclado a memoria, si hay alguno disponible. El caracter no es visualizado.

$KBWAIT 73/X'0049'

Aguarda por un caracter en el teclado. Si se presiona BREAK, este se transfiere como un caracter cualquiera. El caracter teclado no se visualiza.

$KBLINE 64/X'0040'

Espera por una línea completa, terminada con RETURN y la regresa como resultado. Los caracteres tecleados si se visualizan.

$KBBRK 653/X'028D'

Busca unicamente por la tecla BREAK y la regresa como resultado.

- SALIDA A IMPRESORA.

$PRCHAR 59/X'0038'

Transfiere un caracter a la impresora, si esta no esta disponible espera a que lo esté.

$PRSCN 473/X'01D9'

Esta rutina copia los 1023 caracteres de la pantalla a la impresora, si la impresora no está disponible espera a que lo esté.

- SALIDA A LA PANTALLA DE VIDEO.

$VDCHAR 51/X'0033'

Visualiza un caracter en la posición actual del cursor en la pantalla.

$VDCLS 457/X'01C9'

Limpia completamente la pantalla de video.

$VDLINE 539/X'021B'

Visualiza una línea completa en pantalla. La línea debe terminar con un "retorno de carro" (X'0D') ó ETX(X'03'); el primero es impreso, el segundo no.

ENTRADA/SALIDA DE LA INTERFASE RS232C.

$RSINIT 90/X'005A'

Inicializa la interfase RS232C.

$RSRCV 80/X'0050'

Recibe un caracter de la interfase RS232C.

$RSTX 85/X'0055'

Transmite un caracter a la interfase RS232C.

VI. DIRECCIONES INPORTANTES DE RAM.

Colocando varios valores en las direcciones listadas abajo, se pueden activar o controlar muchas de las posibilidades de TRS-80 Modelo III. Para más información de su uso, consultar la función POKE o ver las páginas 189 a 190 del Manual de Referencia.

A continuación se dan direcciones de memoria, tanto decimal como hexadecimal, uso y contenido inicial de ellas.

16409/X'4019' Para alternar mayúsculas y minúsculas. Colocando un 0 tenemos minúsculas y mayúsculas; contenido diferente de 0 permite solo mayúsculas. Contenido inicial "Mayúsculas".

16412/X'401C' Control del cursor. Con un 0 el cursor es intermitente diferente de cero el cursor permanece fijo. Contenido inicial intermitente.

16416/X'4020' Dirección del cursor. Para situar al cursor en determinada posición.

16419/X'4023' Código del caracter ASCII que representa al cursor. Inicialmente es el 176.

16424/X'4028' Número máximo de líneas por página más uno en la impresora. Inicialmente 67.

16425/X'4029' Número de líneas impresas más uno. Inicialmente 1.

16427/X'402B' Máxima longitud de línea en la impresora menos dos. El número máximo es 255.

16526/X'408E' Dirección de la rutina USR, abarca dos bytes: 16526 y 16527. Inicialmente contiene el 7754.

16913/X'4211' Para seleccionar velocidad de transferencia de datos; teniendo 0 tenemos 500 Baud, diferente de cero 1500 Baud.

16916/X'4214' Protección de las 7 líneas superiores de la pantalla; puede tener de 0 al 7 y valores mayores se interpretan como módulo 8.

16919/X'4217' Contiene la fecha y la hora en seis bytes. Del byte 16919 al 16924, se tienen respectivamente segundos, minutos, hora, año, día y mes.

16928/X'4220' Se utiliza en la subrutina $ROUTE y contiene el dispositivo destino en dos bytes, 16928 y 16929.

16930/X'4222' Se utiliza en la subrutina $ROUTE y contiene el dispositivo fuente en dos bytes, 16930 y 16931.

VII. CODIGOS DE CONTROL DE VIDEO.

Consultar la página 228 del Manual de Referencia. A continuación se dan los códigos que tienen efecto en TRS-80 Modelo III. El código se da en decimal y hexadecimal.

8/X'08'	Regresa el cursor y borra el caracter.
9/X'09'	Avanza el cursor al siguiente campo de tabulación.
10/X'0A'	Mueve el cursor al inicio de la siguiente línea, borra la línea y provoca un "retorno de carro".
13/X'0D'	Actúa igual que 10/X'0A'
14/X'0E'	Prende el cursor. En impresora se utiliza para imprimir caracteres de doble ancho.
15/X'0F'	Apaga el cursor.
21/X'15'	Alterna los caracteres especiales/compresión.
22/X'16'	Alterna los caracteres especiales.

23/X'17'	Cambia a caracteres de doble ancho.
24/X'18'	Regresa el cursor sin borrar.
25/X'19'	Avanza el cursor.
26/X'1A'	Baja el cursor.
27/X'1B'	Sube el cursor. Al usar la impresora equivale a ESCAPE.
28/X'1C'	Coloca al cursor en la esquina superior izquierda.
29/X'1D'	Regresa el cursor al principio de la línea y borra la misma.
30/X'1E'	Borra hasta el fin de la línea.
31/X'1F'	Borra hasta el fin de la pantalla.

VIII. MENSAJES DE ERROR. Código.

Ya sea para identificar un error al correr un programa o para fabricar rutinas de manejo de error, se dan los siguientes códigos, abreviaciones y significados. Consultar el Manual de Referencia, páginas 223 a 225, para más información.

CODIGO	ABREVIATURA	SIGNIFICADO
1	NF	Se encontró un NEXT sin haberse registrado un FOR.
2	SN	Error de sintaxis.
3	RG	Se encontró un RETURN sin registrarse antes un GOSUB.
4	OD	No existen datos para ejecutar un READ o un INPUT#.
5	FC	Se intenta usar una función utilizando un parámetro ilegal.
6	OV	Overflow. La magnitud de un número leído o derivado es más grande de lo que la computadora puede manejar.
7	OM	Toda la memoria disponible ha sido usada o reservada.
8	UL	Se intenta referir o brincar a una línea que no existe.

CODIGO	ABREVIATURA	SIGNIFICADO
9	BS	Se intenta asignar o hacer referencia a un elemento con un indice más grande que el dimensionado con la instrucción DIM.
10	DD	Se intenta dimensionar una variable que anteriormente fué ya dimensionada.
11	/0	Se intenta hacer una división con un divisor igual a cero.
12	ID	Uso ilegal de un comando, como INPUT en modo directo.
13	TIM	Se intenta hacer una asignación con operadores incompatibles, esto, es un string con un no string y viceversa.
14	OS	Fuera del espacio localizado para una variable de tipo "string".

CODIGO	ABREVIATURA	SIGNIFICADO
15	LS	Se intenta asignar a una variable string un valor que excede de 255 caracteres.
16	ST	Una operación con "strings" fué demasiado compleja. Dividase en pasos más cortos.
17	CN	El programa no puede continuar.
18	NR	No se puede ejecutar un RESUME.
19	RW	Se encontro una proposición RESUME sin registrarce antes un ON ERROR GOTO.
20	UE	Se inentó generar un error con un código ilegal.
21	MO	Falta un operando al ejecutar una instrucción.
22	FD	Existe error en la entrada de un archivo de datos de una fuente externa (cassettera).

CODIGO	ABREVIATURA	SIGNIFICADO
23	L3	Se trata de utilizar una instrucción solo permisible en sistema de DISCO.

IX. CARACTERES ESPECIALES.

' Abreviatura de REM

% Hace variables de precisión entera.

! Hace variables de precisión simple.

Hace variables de precisión doble.

$ Hace variables tipo "string".

: Separa instrucciones en una misma línea.

? Igual que PRINT (no es válido L? por LPRINT)

, Para escribir cada espacio de tabulación.

; Para escribir sin dejar espacio.

X. OPERADORES.

En orden de procedencia:

↑ ó [Exponenciación

- , + Signo unario negativo y positivo.

* , /	Multiplicación, división.
+ , -	Suma y concatenación, resta.
<,>,=, <=,>=,<>	Operadores relacionales.
NOT	
AND	
OR	

DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA U.N.A.M.

LENGUAJE DE PROGRAMACION BASIC "PRIMERA PARTE"

P R O G R A M A C I O N E S T R U C T U R A D A C O N

B A S I C T R S - 80

CONCEPTO DE UNA COMPUTADORA

Febrero, 1983

Palacio de Minería Calle de Tacuba 5 primer piso Deleg. Cuauhtemoc 06000 México, D.F. Tel.: 521-40-20 Apdo. Postal M-2285

UNAM

CONCEPTO DE UNA COMPUTADORA

FACULTAD DE INGENIERIA

A TOMADO DE: "QUE HACEN LAS COMPUTADORAS Y COMO LO HACEN"

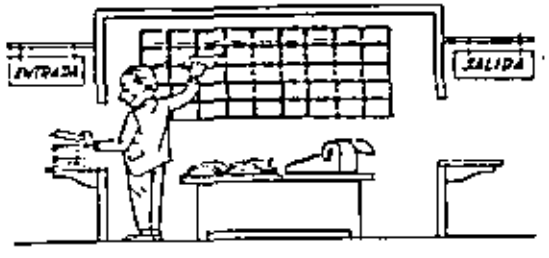
POR: I B M, Argentina.

CONCEPTO DE COMPUTADORA

El objeto de esta breve reseña sobre las computadoras electrónicas y sus múltiples aplicaciones al servicio del hombre, es transmitir al lector una completa visión de conjunto, mediante un lenguaje sencillo que permita comprender conceptualmente los temas tratados, sin necesidad de conocimientos previos en la materia.

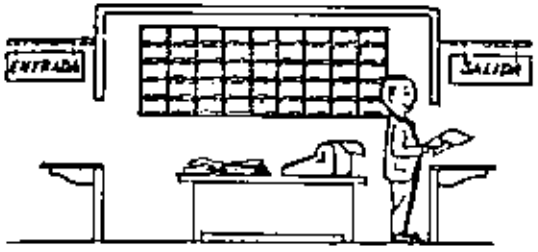
Esperamos que estas páginas, muy simples en apariencia pero con profundo contenido, permitan, a quienes las lean, ingresar al maravilloso mundo de las máquinas automáticas.

Este señor se llama Control. Trabaja en una pequeña habitación. Tiene a su disposición una máquina de calcular que suma, resta, multiplica y divide. Tiene también el señor Control un archivo parecido al casillero que existe en los trenes para clasificación postal.
Hay, además, en la habitación, dos ventanillas identificadas con sendos carteles: "Entrada" y "Salida".
El señor Control tiene un manual que le indica cómo debe desenvolverse con estos elementos, si alguien le pide que haga un trabajo.

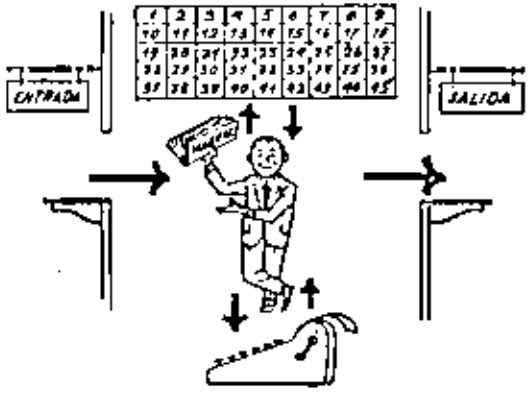


Una persona quiere saber el resultado de un complicado cálculo. Para ello, escribe ordenada, precisa y detalladamente, cada una de las operaciones que, en conjunto, integran ese cálculo, anota cada instrucción elemental en una hoja de papel y coloca todas las hojas en orden en la ventanilla "Entrada".
El señor Control, al ver las hojas, lee en su manual que debe tomar esas hojas con instrucciones, una por una, y colocarlas correlativamente en su archivo. Y así lo hace.

Una vez ubicadas todas las instrucciones en el archivo, el señor Control consulte nuevamente el manual. Allí se le indica que, a continuación, debe tomar la instrucción de la casilla 1 y ejecutarla, luego la de la casilla 2 y ejecutarla, y así sucesivamente hasta ejecutar la última instrucción. Algunas instrucciones indicarán que hay que sumar una cantidad a otra (instrucciones aritméticas); otras, que el señor Control debe ir a la ventanilla "Entrada" para buscar algún dato que intervenga en el cálculo -- (instrucciones de "entrada/salida"), dato que la persona que le formuló el problema habrá colocado ya en dicha ventanilla, en otra hoja de papel.
Finalmente, otras instrucciones indicarán que debe elegirse una de entre dos alternativas (instrucciones lógicas): por ejemplo, supongamos que una parte del cálculo - desde la instrucción que está en la casilla 5 del archivo hasta la que está en la casilla 9 debe ejecutarse 15 veces porque el cálculo así lo exige.
En tal caso, la instrucción que está en la casilla 10 indicará - que, si los pasos 5 a 9 se han ejecutado menos de 15 veces, se - debe volver al paso 5. Cuando se hayan realizado las 15 repeti - ciones y no antes, el señor Control seguirá con la instrucción - de la casilla 11.



Después de ejecutar todas las instrucciones del archivo, haciendo con la máquina de calcular las operaciones en ellas indicadas, el señor Control entrega, a través de la ventanilla "Salida", los resultados obtenidos . . . y se sienta a esperar un nuevo trabajo.



Obsérvese que la actuación del señor Control es puramente mecánica: sólo sigue las indicaciones de su manual y cumple de acuerdo con - ellas, las instrucciones que recibe a través de la ventanilla "Entrada" . Toma decisiones, pero solamente cuando se le señalan las alternativas que existen y con qué criterio debe elegir una de - ellas.
El señor Control puede resolvernos cualquier problema, por complicado que éste sea. Pero para ello debemos indicarle paso a paso, en - la forma más elemental y detallada, todo lo que debe hacer para resolverlo, sin olvidarnos absolutamente nada porque, en ese caso, el señor Control no sabría continuar por sí mismo.
Haga el lector la prueba de formular un problema cualquiera de modo tal que una persona que no conozca nada acerca de ese problema, pueda resolverlo sin necesidad de hacer consultas. Verá que es una experiencia interesantísima.

El esquema que acabamos de representar mediante el señor Control y sus - elementos de trabajo, corresponde exactamente al esquema de funcionamiento de una computadora electrónica.

A continuación presentaremos una breve descripción de los elementos de - la computadora que corresponden a los elementos de trabajo del señor Control.

Las unidades de Entrada (representadas por la ventanilla "Entrada") : -

Son en la computadora, dispositivos capaces de leer información (Instrucciones o Datos) con el objeto de procesarla. Existen una gran variedad de elementos de entrada, entre los cuales tenemos:

Tarjetas de Cartulina y Cintas de Papel: Que son perforadas de manera que cada perforación representa un número, una letra ó un símbolo especial de acuerdo con un código predeterminado.

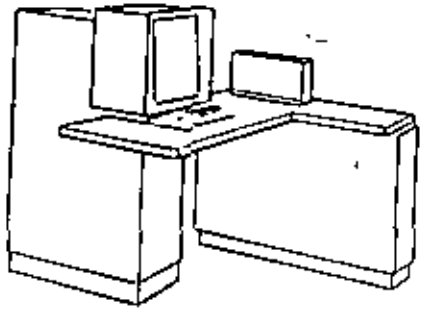
Cintas magnéticas: Conocidas como "memorias externas" tienen la ventaja - de permitir almacenar la información en forma mas concentrada (a razón - de 80 a 2400 caracteres por pulgada de longitud) y de ser mas veloces, - ya que pueden enviar o recibir información a la unidad de control a velocidades que van de 10,000 a 680,000 caracteres por segundo. Pueden llegar a tener hasta 730 m. de longitud.

Disco Magnético: También conocidos como "Memoria externa", en general tienen un diámetro aproximado de 30 cm. y pueden grabar hasta 400,000 letras, números, y caracteres especiales, formando palabras, cifras, ó registros - completos. Se pueden grabar o leer a razón de 77,000 a 312,000 caracteres por segundo y su tiempo de acceso a un registro alcanza un promedio de 60 milí-segundos.

Una diferencia importante entre las cintas y los discos es la siguiente:
En las cintas los registros se graban o leen secuencialmente..
En los discos se tiene "Libre Acceso" a un registro cualquiera, en forma inmediata, pues cada registro se localiza por su posición física dentro del disco.

Lectora Optica de Caracteres Impresos: Puede leer un documento impreso por una máquina de escribir, o por una máquina de contabilidad o por la impresora de una computadora, a una velocidad de 30,000 caracteres por minuto.

Unidad de Representación Visual: Esta unidad de entrada/salida sirve para hacer consultas a la computadora, por medio de un teclado de máquina de escribir, y obtener la respuesta reflejada en una pequeña pantalla de televisión.
La imagen está formada por hasta 12 renglones de hasta 80 caracteres (letras, numeros, ó signos especiales) cada uno.



Vemos aquí otra Unidad de Representación Visual, más evolucionada que la anterior, la comunicación hombre-máquina puede establecerse en ella por medio de gráficas, es decir que la entrada y la salida de datos se hacen por medio de imágenes.

Cuenta esta unidad para ello con un dispositivo con forma de lápiz, que tiene en su punta una célula fotoeléctrica. Un delgado haz de luz parte en determinado momento de un punto de la pantalla y la recorre en forma de zig-zag. Si se apoya el "lápiz" en cualquier posición de la pantalla, su célula fotoeléctrica detectará en algún momento el haz de luz.

Por el tiempo transcurrido desde que el haz de luz comenzó su "barrido" hasta que fue detectado, la computadora determina en qué punto de la pantalla se encuentra apoyado el "lápiz" .

Como el barrido dura una fracción de segundo y se realizan muchos barridos por segundo, se puede "escribir" con el "lápiz" sobre la pantalla y el dibujo "ingresa" en la memoria de la computadora como una sucesión de puntos codificados.

La pantalla está imaginariamente dividida en 1.040.576 puntos, de manera que los trazos que se obtienen son prácticamente continuos.
Pueden dibujarse así curvas, estructuras, letras, números y cualquier tipo de gráfico, y esa información ingresa automáticamente a la computadora.

Por otra parte, los resultados obtenidos por la computadora son representados en la pantalla también como curva, letras, etc., bajo control del programa almacenado en la memoria.

Lectora Optica de Manuscritos: Salvo algunas pequeñas restricciones en cuanto al formato de los caracteres, esta unidad puede "leer" documentos escritos por cualquier persona y con cualquier ejemplo a una velocidad aproximada de 30,000 caracteres por minuto.

El registrador/analizador Fotográfico es una Unidad de Entrada/Salida de datos que realiza las siguientes funciones.

1) Registra los resultados de la computadora sobre microfotografías, mediante un tubo de rayos catódicos, que inciden sobre una película fotográfica, y cuyo haz electrónico actúa gobernado por el Programa Almacenado. La película se revela automáticamente dentro de la unidad y 48 segundos después está lista para ser proyectada.

2) Proyecta sobre una pantalla translúcida las microfotografías registradas.

3) Analiza imágenes reproducidas en negativo sobre película transparente, las digitaliza y las transmite a la Unidad Central de Procesamiento.

La película utilizada tiene 30.5 milímetros de ancho y 120 metros de longitud. La Entrada o Salida de imágenes puede consistir en letras, números, símbolos, dibujos, gráficas, mapas, curvas, etc. En una microfotografía de 30.5 mm X30.5 mm pueden registrarse hasta 30,600 letras y números, o hasta 16,777,216 puntos correspondientes a imágenes.

La velocidad de Registro/Análisis es de 40,000 letras, números y símbolos por segundo, o su equivalente si se trata de imágenes.

Máquina de Escribir (Teletipo).

Las unidades de almacenamiento o memorias (Representadas por el archivo del señor Control) permiten registrar las instrucciones y los datos para resolver un problema; entre estas se tienen:

Los Anillos Magnetizantes: Estos pueden magnetizarse en un sentido ó en otro "Recordando" así un 1 o un 0 respectivamente. Con 8 de éstos anillos se forma una posición de memoria, en la cual puede registrarse una letra, un dígito ó un carácter especial, según las distintas combinaciones de anillos "En 1" y "En 0", de acuerdo a un código predeterminado.

Las Memorias de flip -flops

Las Cintas Magnéticas

Los Discos Magnéticos

El dispositivo aritmético (representado por la máquina de calcular) que realiza las cuatro operaciones aritméticas.

Las unidades de salida (representadas por la ventanilla "Salida" que pueden ser:

Impresoras

Máquinas de Escribir (Teletipos)

Grabadoras de Cintas Magnéticas

Grabadoras de Discos Magnéticos

Unidad de Representación Visual

Registrador Analizador Fotográfico

Unidad de Respuesta Oral con la cual la Computadora puede hablar en todo el sentido de la palabra.

Contiene una cinta magnetofónica en la cual un locutor ha grabado un diccionario de una gran variedad de palabras, en cualquier idioma.

Enunciaremos brevemente los adelantos que esta tercera generación ha introducido con respecto a la tecnología anterior :

. La computadora se autogobierna y trabaja sin detenerse, pasando de un trabajo a otro sin demora alguna.

. El Operador interviene sólo cuando algún problema excepcional ocurre. La comunicación entre hombre y máquina se realiza sólo sobre la base de "Informes por Excepción" .

. Si ocurre una falla en los circuitos o en la parte electromecánica la máquina realiza un autodiagnóstico e indica cuál es la anomalía.

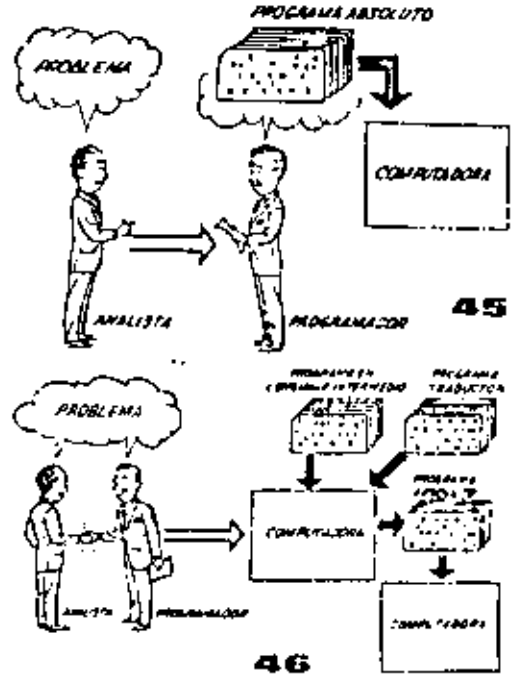
. La velocidad de Entrada-Proceso-Salida se ha incrementado extraordinariamente.

.Todas las operaciones del sistema se realizan en forma simultánea.

. Los lenguajes de programación han evolucionado de manera notable.

. El autocontrol y la autoverificación de operaciones han alcanzado niveles insospechados.

. Pueden realizarse, con máximo rendimiento, varios trabajos distintos simultáneamente. (Multiproceso).

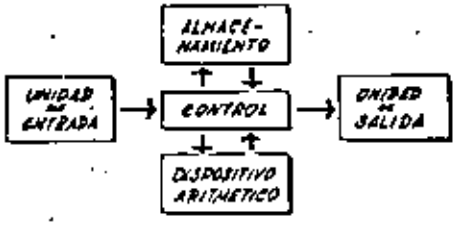


Hasta ahora hemos visto muchas unidades que, en distintas combinaciones, configuran computadoras electrónicas para las más variadas aplicaciones. Ahora nos detendremos para analizar el manejo de dichos sistemas.

El Programa de Instrucciones almacenado en la Unidad Central de Procesamiento, consta de una secuencia de órdenes y comandos, expresados según una codificación especial denominada "Lenguaje Absoluto de Máquina". Las primeras computadoras se "programaban" en este complejo lenguaje. Había entonces una enorme diferencia entre nuestro idioma y aquél según el cuál debíamos comunicarnos con la máquina. Esto obligaba a un gran esfuerzo común entre el analista que conocía el problema, y el programador que conocía la computadora, pues ambos hablaban del mismo proceso en distintos lenguajes.

Finalmente, un dispositivo electrónico de control (representado por - el señor control) ayudado de un programa especial o sistema operativo (repre - sentado por el manual del señor Control), gobierna todas las operaciones de to das las unidades que componen la computadora.

Habiendo descrito las partes que componen la computadora podemos mostrar el si - guiente esquema que la representa:



O en forma más resumida :

S i e n d o :

UNIDAD CENTRAL { ALMACENAMIENTO, CONTROL, DISPOSITIVO ARITMETICO

Hemos hablado hasta este momento de la computadora electrónica desde el punto de vista conceptual. Durante las dos últimas décadas se han producido avances tecnológicos tan extraordinarios en materia de electrónica que la computadora ha sufrido enormes transformaciones. Veremos ahora cómo se ha ido modificando la idea original hasta llegar a los más modernos sistemas de procesamiento de datos.

Las primeras computadoras tenían circuitos con válvulas de vacío. Los tiempos de operación se medían en ellas en milisegundos (milésimas de segundo). Cuando aparecieron los transistores, el diseño de los circuitos se mejoró notablemente y la duración de las operaciones en las computadoras que utilizaban esta "Tecnología de Estado Sólido", se midió en microsegundos (millonésimas de segundo) .

El hecho de que las nuevas máquinas fueran miles de veces más rápidas que las anteriores, trajo aparejada la creación de unidades de entrada, salida y memoria externa mucho más veloces.

La invención de un nuevo tipo de transistor ("chip") provocó una verdadera revolución en los circuitos electrónicos y sus procesos de fabricación. El nuevo elemento es tan pequeño que en un dedal de costura caben más de 50,000 chips. Debido a su tamaño, se les denomina circuitos microminiaturizados o microcircuitos. Los tiempos de operación se miden ahora en nanosegundos (milmillonésimas de segundo). Ha nacido en esta forma la tercera generación de computadoras, y las altas velocidades alcanzadas posibilitaron un nuevo enfoque en el diseño de los sistemas de procesamiento de datos.

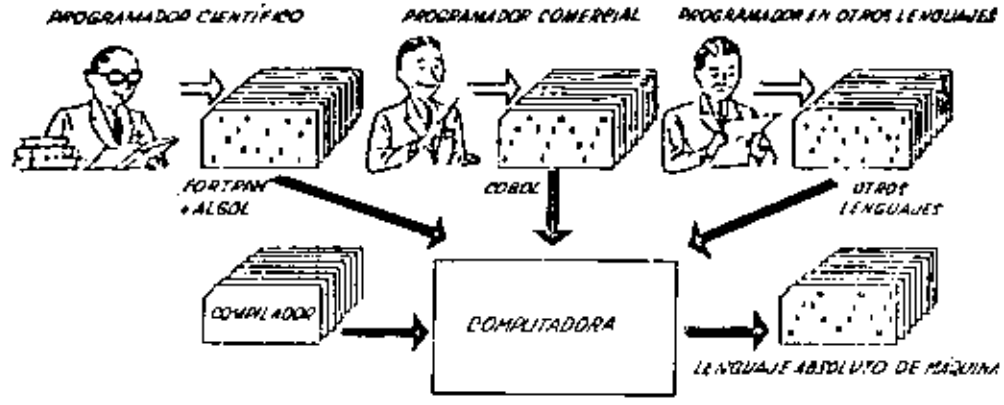
Se crearon, para solucionar el problema, lenguajes intermedios cada vez más parecidos a nuestro idioma. Es decir que cada nuevo lenguaje intermedio se acercaba más al problema y se alejaba más de la máquina. Para cada uno de estos lenguajes se creó un programa traductor llamado "Compaginador" o "Compilador", que tenía la misión de traducir el lenguaje intermedio al absoluto de máquina. Ahora, el analista y el programador "hablan un mismo idioma" : ambos conocen el problema y la solución.

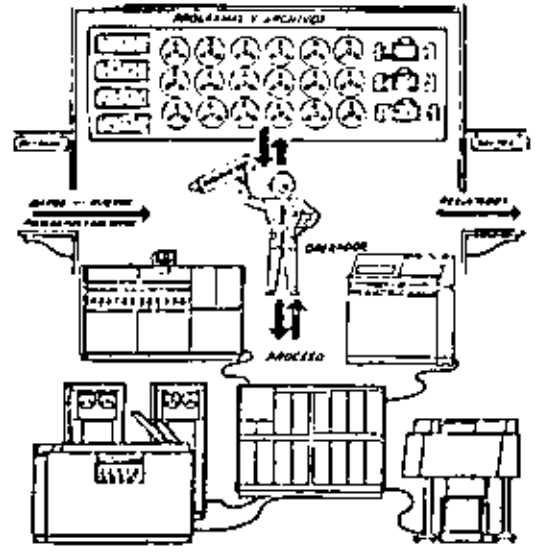
Pero la computadora seguía desarrollándose, y pronto los lenguajes intermedios fueron insuficientes para formular intrincados problemas científicos o comerciales. Nacieron, entonces, lenguajes especializados: dos de ellos, el FORTRAN y el ALGOL, permiten programar problemas científicos+técnicos utilizando una notación casi idéntica a la notación matemática común. El COBOL es un lenguaje comercial cuyas sentencias configuran oraciones y frases en forma tal que una persona que no sabe qué es una computadora, puede leer un programa y entender perfectamente qué es lo que hará la máquina cuando lo tenga almacenado.

Cada uno de estos lenguajes tiene un programa Compilador para cada tipo distinto de computadora capaz de procesarlo. Esto significa que un programador que sabe FORTRAN, por ejemplo, puede programar una computadora aún sin conocerla. Es decir que estos tres lenguajes constituyan un "esperanto" de las máquinas.

La tercera generación de computadoras permitió abordar complejos problemas que incluían, entre otros, aspectos comerciales y científicos.

Hemos llegado así a que la computadora nos "entiende", en lugar de que se limite a recibir órdenes en su idioma.





DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA U.N.A.M.

LENGUAJE DE PROGRAMACION BASIC CON APLICACIONES

(PRIMERA PARTE)

Q U I C K

R A D I O S H A C K

SISTEMA TRS - 80 MODELO III

FEBRERO, 1983

Palacio de Minería Calle de Tacuba 5 primer piso Deleg. Cuauhtemoc 06000 México, D.F. Tel.: 521-40-20 Apdo. Postal M-2285

Q U I C K

R A D I O S H A C K

SISTEMA TRS-80 MODELO III

M I C R O C O M P U T A D O R A R A D I O S H A C K

S I S T E M A T R S - 8 0 M O D E L O I I I

INICIALIZACION.

El sistema completo (la computadora y los periféricos) debe estar apagado.

1. Primeramente encender todos los periféricos y después la computadora.

2. El mensaje: Cass?
debe aparecer en la pantalla. Para seleccionar una alta velocidad de grabación en cassette (1500 bauds), presione la tecla H o ENTER . Para seleccionar una baja velocidad de grabación en cassete (500 bauds), presione la tecla L .

   Para propósitos generales, use una velocidad de grabación alta. Para salvar o cargar un programa en Basic Nivel II Modelo I, se debe usar la velocidad de grabación baja.

3. El mensaje: Memory Size?
debe aparecer en la pantalla. Para utilizar toda la memoria disponible, presione la tecla ENTER . Para reservar algo de espacio en memoria, teclee la dirección más alta (en decimal) que usted quiera usar, entonces presione la tecla ENTER .

4. El mensaje:
```
Model III Basic
(c) Tandy'80
READY
>
```
debe aparecer en la pantalla. Entonces la computadora está lista para utilizarse.

INSTRUCCIONES.

- AUTO inicio, incremento
  El número de línea aparecerá automáticamente.
  AUTO AUTO 150 ,20 AUTO ,5

- CLEARn
  Asigna n bytes para cada variable tipo string; inicializa todas las variables.
  CLEAR CLEAR 75 CLEAR 0

- CLOAD
  Para cargar en la memoria de la computadora un programa en Basic salvado en cassette. Si se especifica un nombre sólo se tomará en cuenta la primera letra.
  CLOAD CLOAD "PRUEBA"

- CLOAD?
  Compara un programa en cinta magnética, byte por byte, con el programa residente en la computadora.
  CLOAD? CLOAD? "PRUEBA"

- CLS
  Limpia la pantalla.
  CLS

- CONT
  Continúa la ejecución de un programa después de una interrupción por una instrucción STOP o por presionar la tecla BREAK .
  CONT

- CSAVE
  Salva un programa residente en cinta magnética. Se debe especificar un nombre. Sólo el primer caracter del nombre es usado.
  CSAVE "PRUEBA"

- DATA
  Guarda datos que serán llamados por una instrucción READ.
  DATA "HERNANDEZ,V.",1960

- DEFDBL
  Define variables de doble presición.
  DEFDBL V,X-Z

- DEFINT
  Define variables de tipo entero.
  DEFINT A,I-N

- DEFSNG
  Define variables de presición simple.
  DEFSNG I,W-Z

- DEFSTR
Define variables de tipo string.
DEFSTR C,L-Z

- DELETE
Borra líneas de un programa.
DELETE 1205 DELETE -80 DELETE 100-300

- DIM
Dimensiona uno o más arreglos.
DIM R(75),W(40) DIM L$(3,18,5) DIM AR$(8,25)

- EDIT
Pone a la computadora en modo editor para la línea especificada. Vea comandos del modo editor.
EDIT 100 EDIT.

- END
Finaliza la ejecución del programa.
END

- ERROR(n)
Simula el error especificado entre paréntesis de acuerdo al código de mensajes de error ($1 \le n \le 23$).
ERROR(1)

- FOR...TO...STEP/NEXT
Expresión iterativa.
FOR J=1 TO 8 (...) NEXT J
FOR C!=0 TO 5 STEP 2 (...) NEXT C!

- GOSUB
Transfiere el control a la subrutina indicada.
GOSUB 750

- GOTO
Transfiere el control a la línea indicada.
GOTO 180

- IF...THEN...ELSE
Expresión condicional.
IF P=Q THEN 200
IF N%<0 THEN 150 ELSE N%=N%-1

- INPUT
Para lectura de datos en pantalla.
INPUT X# INPUT L,M,N INPUT "DATO";N

- INPUT#-1
Para lectura de datos desde cassette.
INPUT#-1,NUM

- LET
Asigna un valor a una variable (opcional).

LET X=7.05 LET R2=RI LET C$="ROJO"

- LIST
Lista las líneas del programa en la pantalla.
LIST LIST 50-85

- LLIST
Lista las líneas del programa en la impresora.
LLIST LLIST 50-

- LPRINT
Imprime en la impresora letrero(s) y/o variable(s).
LPRINT CAP$,"ES EL CAPITAL DE";ST$

- LPRINT TAB
Mueve el carro de la impresora a la posición especificada.
LPRINT TAB(25)"REPORTE"

- LPRINT USING
Imprime con el formato especificado, numeros y strings en la impresora. Vea PRINT USING para opciones de esta instrucción.
LPRINT USING "####.";1234

- NEW
Borra el programa en memoria; inicializa todas las variables.
NEW

- ON ERROR GOTO
Si hay un error transfiere el control hasta la línea especificada.
ON ERROR GOTO 2100

- ON ERROR GOTO 0
Inutiliza la rutina de error.
ON ERROR GOTO 0

- ON...GOSUB
Transfiere el control a alguna de las subrutinas especificadas dependiendo del valor de la variable.
ON Y GOSUB 50,100,150,200

- ON...GOTO
Transfiere el control a alguna de las líneas especificadas dependiendo del valor de la variable.
ON X GOTO 190,200,210

- OUTp, v
Envía el valor especificado en v a la localidad p ($0 \le p \le 255$).
OUT 255, 0

- POKE n, v
Pone el valor v(0<v<255) en la localidad n (desde 15360 hasta el fin de la memoria). Vea direccionamiento del POKE.
PRINT 15872, 255

- PRINT
Imprime letreros y/o variables en la pantalla.
PRINT X;Y; PRINT "MEXICO"

- PRINT@n
Imprime empezando en n(0<n<1023)
PRINT@477, "CENTRO"

- PRINT#-1
Graba datos en cinta magnética.
PRINT#-1, A

- PRINT TAB
Mueve el cursor hacia la derecha hasta la posición indicada.
PRINT TAB(20) "REPORTE"

- PRINT USING
Formatos para números y string.

Formato para números.
PRINT USING "#####"; 66.2

. Punto decimal
PRINT, USING "##.#"; 58.76

, Aparecerá una coma cada tercer dígito.
PRINT USING "####,"; 1234

** Llena los primeros espacios con asterisco.
PRINT USING "**####"; 44.0

$$ Signo flotante de pesos.
PRINT USING "$$##.##"; 118.6735

**$ Signo flotante de pesos; llena los primeros espacios con asteriscos.
PRINT USING "**$#.##"; 8.333

^^^^ Formato para exponenciales. Presione la tecla ↑ para que aparezca este caracter.
PRINT USING "###.#^^^^"; 8527100

+ En la primera posición provoca que el signo sea impreso; en la última posición provoca que el signo sea impreso después del número.
PRINT USING "+###"; -216

- Imprime el signo menos después de un número negativo, si es un número positivo deja un espacio en blanco.
PRINT USING "####.##-"; -8124.42

! Imprime el primer caracter del string.
PRINT USING "!"; "HOLA"

%espacios%
Campo de un string; la longitud de campo es el número de espacios más dos.
PRINT USING "% %"; "AZUL"

- RANDOM
Genera números en forma aleatoria.
RANDOM

- READ
Toma valores de una instrucción DATA.
READ T READ S$ READ NM$,EDAD

- REM
Comentario; instrucción que indica a la computadora que debe ignorar el resto de la línea. El apóstrofe (') es una abreviación de REM.
REM ESTA LINEA ES UN COMENTARIO 'ESTA TAMBIEN

- RESET (x,y)
Apaga el punto especificado de la gráfica.
x: eje horizontal (0<x<127). y: eje vertical (0<y<47).
RESET (21, 40) RESET (L1, L2)

- RESTORE
Cuando una instrucción READ aparece después de una instrucción RESTORE, entonces se volverán a utilizar los mismos datos de la primera instrucción DATA.
RESTORE

- RESUME
Termina la rutina de error y ejecuta la línea indicada.
RESUME 40 RESUME RESUME NEXT

- RETURN
Transfiere el control a la siguiente instrucción después de un GOSUB.
RETURN

- RUN
Ejecuta el programa residente o una porción de él.
RUN RUN 500

- SET (x, y)
Prende el punto indicado.
x: eje horizontal (0<x<127). y: eje vertical (0<y<47).
SET (10,0) SET (L1, L2)

- STOP
Detiene la ejecución de un programa.
```
STOP
```

- SYSTEM
Pone a la computadora en modo monitor para cargar en disco archivos en lenguaje de máquina. A lo cual responderá con un *?, entonces teclee el nombre del archivo o /dirección.
```
SYSTEM
```

- TROFF
Anula la función TRACE.
```
TROFF
```

- TRON
Prende la función TRACE.
```
TRON
```


C O D I G O D E C O N T R O L D E V I D E O

DECIMAL	HEXADECIMAL	PRINT CHR$ (código)
8	08	Espacios en blanco y borra el caracter actual.
10	0A	Imprime una línea y regresa el cursor
13	0D	Imprime una línea y regresa el cursor.
14	0E	Aparece el cursor.
15	0F	Desaparece el cursor.
21	15	Switchea caracteres especiales y compresión.
22	16	Switchea alternativamente los caracteres.
23	17	Cambia a modo de 32 caracteres por línea.
24	18	Mueve el cursor a la derecha sin borrar.
25	19	Avanza el cursor.
26	1A	Baja una línea.
27	1B	Sube una línea.
28	1C	Coloca el cursor al inicio de la pantalla.
29	1D	Mueve el cursor al inicio de la línea.
30	1E	Borra toda la línea.
31	1F	Limpia al final de la pantalla.

O P E R A D O R E S

Cada operador o grupo de operadores tiene menor prioridad que el anterior en el orden siguiente:

| o [— Exponenciación (para precisión simple). Presione la tecla ↑ para que aparezca este operador; deberá aparecer el paréntesis cuadrado izquierdo.

- , + — Negativo unario, positivo.

* , / — Multiplicación, división.

+ , - — Suma y concatenación (string), substracción.

<, >, =, <=, >=, <> — Comparaciones.

NOT

AND

OR

CARACTERES ESPECIALES

.	Abreviación de la instrucción REM.
%	Para variables enteras.
!	Para variables de precisión simple.
#	Para variables de doble precisión.
$	Para variables de tipo string.
:	Separa instrucciones en una misma línea.
?	Abreviación de la instrucción PRINT (no es válido LI en vez de LPRINT).
,	Puntuación en la instrucción PRINT; para impresión en zonas de 16 columnas.
;	Puntuación de la instrucción PRINT, para separar letreros y/o variables en la declaración de la instrucción, pero en la impresión no separa con espacios.

COMANDOS EN MODO EDITOR

A	Anula los cambios y empieza otra vez.
n C	Cambia n caracteres.
n D	Borra n caracteres.
E	Sale del modo EDIT y actualiza los cambios realizados.
H	Corta la línea e inserta al final.
I	Inserta caracteres.
n K c	Borra todos los caracteres después de la n-ésima ocurrencia de c.
L	Lista la línea.
Q	Se desocupa el modo editor y cancela todos los cambios.
n S c	Busca la ocurrencia de c, tantas veces el valor de n.
X	Inserta al final de la línea.
SHIFT ↑	Sale del modo editor.
ENTER	Reconoce los cambios, permaneciendo en modo editor.
n SPACERBAR	Mueve el cursor n espacios a la derecha.
n ←	Mueve el cursor n espacios a la izquierda.

TECLAS DE CONTROL

←	Borra el último caracter tecleado, regresa el cursor un espacio.
SHIFT ←	Borra la línea.
BREAK	Interrumpe alguna actividad y regresa al nivel de comandos.
CLEAR	Limpia la pantalla.
ENTER	Comunica el fin de línea.
SPACEBAR	Pone un blanco.
→	Avanza el cursor a la siguiente posición.
SHIFT →	Pone en modo 32-caracteres.
↓	Brinca una línea y pone el cursor.
SHIFT ↓	Tecla de "control". Presione simultáneamente estas dos y alguna tecla A-Z para lograr control A hasta control Z.
SHIFT ↓ *	Copia el contenido de la pantalla a impresora.
SHIFT @	Estatifica la ejecución de un programa (pausa). Para continuar presione alguna tecla.

F U N C I O N E S

Los argumentos estan indicados con letras especiales (variables):

REALES
x: (-1x10E38, -1x10E-38), (1x10E-38, 1x10E38)

ENTEROS
n: (-32768, 32767)

LOCALIDADES
c: (0, 255)

str: argumento tipo string.

var: nombre de variable.

- ABS(x)
Valor absoluto de x.
```
Y = ABS(x)
```

- ASC(str)
Proporciona el equivalente en código ASCII del primer caracter del string.
```
A = ASC(T$)
```

- ATN(x)
Arcotangente de x, en radianes.
```
Y = ATN(x/3)
```

- CDBL(x)
Convierte a la variable x en variable de doble precisión.
```
x# = CDBL(n^3)
```

- CHR$(c)
Proporciona el caracter equivalente en código ASCII, control o código de gráficas.
```
P$=CHR$(T)
```

- CINT(n)
Proporciona el entero más grande no mayor que n.
```
PRINT CINT(15.007$)
```

- COS(x)
Coseno del argumento dado en radianes.
```
Y = COS(x)
```

- CSNG(x)
Convierte a presición simple.
```
FC = CSNG(TH#)
```


- ERL
Proporciona el número de línea en la cual ha ocurrido un error.
```
PRINT ERL
```

- ERR
Si hay un error, proporciona el valor de acuerdo al código de error, este valor será (código de error .1) *2
```
IF ERR=12 THEN 650 ELSE 800
```

- EXP(x)
Calcula el antilogaritmo natural de x.
```
Y = EXP(x)
```

- FIX(x)
Proporciona todos los dígitos a la izquierda del punto.
```
Y = FIX(x)
```

- FRE(número)
Proporciona la cantidad de memoria disponible.
```
F=FRE(x)
```

- FRE(str)
Proporciona el espacio no usado por el string. Str es cualquier consonante o variable tipo string.
```
FRE("c")      FRE(c$)
```

- INKEY$
Proporciona al caracter tecleado, si está disponible.
```
A$ = INKEY$
```

- INP(p)
Proporciona el valor de la localidad p(0≤p≤255).
```
V = INP(255)
```

- INT(x)
Proporciona el entero no mayor que x.
```
Y = INT(x)
```

- LEFT$(str,c)
Proporciona los primeros c caracteres de str.
```
P$ = LEFT$(M$,7)
```

- LEN(str)
Proporciona el número de caracteres del string.
```
X = LEN(SEN$)
```

- LOG(x)
Calcula el logaritmo natural de x.
```
Y = LOG(x)
```

- MEM
Proporciona la cantidad de memoria disponible.
```
PRINT MEM
```

- MID$(str, posición, longitud)
Proporciona un substring de otro string. Si la longitud es omitida, el string a partir de posición será proporcionado.
PRINT MID$(A$,2,3) F$=MID$(A$,3)

- PEEK(n)
Da el valor guardado en la localidad n.
V = PEEK(18520)

- POINT(x, y)
Verifica si el punto de la gráfica está prendido o apagado.
x: eje horizontal (0<x<127). y: eje vertical (0<y<47).
IF POINT(13,35) THEN PRINT "PRENDIDO" ELSE PRINT "APAGADO"

- POS(x)
Proporciona la columna de la posición del cursor (0-63).
X es un argumento dummy.
PRINT TAB(40) POS(0)

- RIGHT$(str, c)
Proporciona la parte derecha del string a partir de c.
ZIP$ = RIGHT$(AD$,5)

- RND(n)
Genera un número aleatorio entre 1 y n, si n>1 o entre 0 y 1 si n=0.
Y=RND(100) PRINT RND(0) R=RND(x)

- SGN(x)
Proporciona el signo: -1, 0, 1 cuando x es negativo, cero o positivo.
x = SGN(A*B)

- SIN(x)
Calcula el seno donde el argumento debe estar en radianes.
Y = SIN(x)

- SQR(x)
Calcula la raíz cuadrada de x.
Y = SQR(A*B)

- STR$(x)
Convierte una expresión numérica en string.
S$ = STR$(x)

- STRING$(l, c)
Proporciona un string de longitud l. Dónde c puede ser en código ASCII o como un string.
B$ = STRING$(125,"?") B$ = STRING$(125,63)

- TAN(x)
Calcula la tangente cuyo argumento es en radianes.
Y = TAN(x)

- TIME$
Proporciona el tiempo (con formato de 24 horas) y el dato es un string de 17 caracteres.
A$ = TIME$

- USR(x)
Llama una subrutina en lenguaje de máquina localizada en la dirección x(16526-16527)
PRINT USR(-1) Y=USR(x)

- VAL(str)
Convierte el string a un número.
V2 = VAL("100 PESOS")

- VARPTR(var)
Proporciona la dirección donde la variable ha sido guardada.
Y = USR(VARPTR(x))

M E N S A J E S D E E R R O R

CODIGO	ABREVIACION	EXPLICACION
1	NF	Next sin su correspondiente FOR.
2	SN	Error de sintaxis.
3	RG	Return sin su correspondiente GOSUB.
4	OD	Faltaron datos en una instrucción READ o INPUT.
5	FC	Llamada incorrecta de una función.
6	OV	Se excedió la capacidad de un número.
7	OM	Memoria saturada.
8	UL	Linea indefinida.
9	BS	Subíndice fuera de rango.
10	DD	Arreglo dimensionado nuevamente.
11	/0	División por cero.
12	ID	Se utiliza la instrucción INPUT como comando directo.
13	TM	Conflicto de operandos.
14	OS	El espacio total de un string fue excedido.
15	LS	String con muchos caracteres.
16	ST	Fórmula del string muy complicada.
17	CN	No puede continuar.
18	NR	No hay una instrucción RESUME.
19	RW	Instrucción RESUME SIN ERROR.
20	UE	Error indefinido.
21	MO	Error de operandos.
22	FD	Archivo de datos mal salvado.
23	L3	Solo para el sistema de Basic con disco.

DIRECCIONAMIENTOS PARA LA INSTRUCCION P O K E

Las siguientes localidades pueden activar o controlar algunas de las características especiales de la TRS-80 Modelo III. Vea el Manual de Operación para mayores detalles.

Por ejemplo:
Para seleccionar una alta velocidad de grabación en cassette, ejecute: POKE 16913,1

LOCALIDAD DEC	HEX	CONTENIDO	CONTENIDO INICIAL
16409	4019	Switch de letras n=0: Letras mayúsculas y minúsculas n≠0: Sólo letras mayúsculas	Mayúsculas
16412	401C	Parpadeo del cursor n=0: Parpadea n≠0: No parpadea	0
16416	4020	Localidad del cursor Dos bytes: LSB, MSB	N/A
16419	4023	Caracter del cursor Código ASCII (0-255)	176
16424	4028	Máximo de líneas/página más uno	67
16425	4029	Número de líneas impresas más uno	1
16427	402B	Longitud máxima de línea menos dos	255
16526	408E	Localidad de la rutina USR Dos bytes: LSB, MSB	7754
16872	41E8	Buffer de lectura $RSRCV Un byte	0
16880	41F0	Buffer de salida $RSTX Un byte	0
16888	41F8	Velocidad de transferencia de código	85
16889	41F9	Código de paridad, Longitud de palabra y Stop-bit	108
16890	41FA	Switch de espera $RSINIT n=0: No espera n≠0: Espera	Espera
16913	4211	Switch de la velocidad de grabación en cassette n=0: 500 bauds n≠0: 1500 bauds	N/A

LOCALIDAD DEC	HEX	CONTENIDO	CONTENIDO INICIAL
16916	4214	Protección de líneas en pantalla, desde 0 a 7. Para valores mayores son interpretados en módulo 8.	0
16928	4220	Dispositivo $ROUTE Dos bytes de E/S designados.	N/A
16930	4222	Dispositivo de default $ROUTE Dos bytes de E/S designados.	N/A

S U B R U T I N A S D E M E M O R I A R O M Z - 8 0

Las siguientes subrutinas ROM pueden ser usadas por un programa en Z-80; algunas de ellas pueden ser usadas en programas de Basic por medio de la función USR. Antes de usar alguna de éstas, lea la sección de información técnica del manual de operación.

LOCALIDAD DEC	HEX	CONTENIDO	FUNCION
0	0000	$RESET	Apaga el sistema.
43	002B	$KBCHAR	Checa el caracter presionado.
51	0033	$VDCHAR	Aparece un caracter.
59	003B	$PRCHAR	Imprime un caracter.
64	0040	$KBLINE	Espera un comando.
73	0049	$KBWAIT	Espera un caracter.
80	0050	$RSRCV	Recibe un caracter desde RS-232-C.
85	0055	$RSTX	Transmite un caracter a RS-232-C.
90	005A	$RSINIT	Inicializa el RS-232-C.
96	0060	$DELAY	Define por un tiempo especificado.
105	0069	$INITIO	Inicializa todos los manejadores de E/S.
108	006C	$ROUTE	Ruta de E/S.
457	01C9	$VDCLS	Limpia la pantalla.
473	01D9	$PRSCN	Imprime el contenido de la pantalla.
539	021B	$VDLINE	Aparece una línea.
565	0235	$CSIN	Lectura de un byte en cassette.
612	0264	$CSOUT	Salida de un byte en cassette.

LOCALIDAD DEC	LOCALIDAD HEX	CONTENIDO	FUNCION
647	0287	$CSHWR	Escribe la cabeza del cassette.
653	0280	$KBBRK	Checa la tecla BREAK.
662	0296	$CSHIN	Lee la cabeza del cassette.
664	0298	$CLKON	Aparece el reloj en la pantalla.
673	02A1	$CKLOFF	Desaparece el reloj de la pantalla.
6681	1A19	$READY	Pasa a Basic, apareciendo "READY".
12339	3033	$DATE	Da la fecha.
12342	3036	$TIME	Da el tiempo.
12354	3042	$SETCAS	Prende la grabadora.
12312	37E8	$PRSTAT	Estado normal de la impresora (realiza lecturas), solo cuando: Bit 7=0 "Está desocupada". Bit 6=0 "Hay papel". Bit 5=1 "Dispositivo seleccionado". Bit 4=1 "La impresora no está saturada". Los bits 3, 2, 1 y 0 no se usan.

PROBLEMA 1) DESGLOSE DE CANTIDADES

ENUNCIADO

Se desea el número de billetes y monedas necesarios para pagar una lista de raya. Se conocen las cantidades a pagar, todos múltiplos de un peso.

SOLUCION

Análisis del problema:

Se utilizarán billetes de 1000,500,100,50 y 20 pesos y monedas de 10,5 y 1 peso.

Se utilizará una denominación solo cuando el monto a desglosar lo exceda.

EJEMPLO:

$50 se descompone en $20+$20+$5+$1+$1+$1+$1

```
10 'DESGLOSE DE CANTIDADES      CARL.  FEB 80
20 INPUT C
25 LPRINTC
30 IF CP < = 0 GOTO 360
40 IF CP <= 1000 GOTO 80
50 CP=CP-1000
60 N1=N1+1
70 GOTO 40
80 IF CP <= 500 GOTO 120
90 CP=CP-500
100 N2=N2+1
110 GOTO 80
120 IF CP <=100 GOTO 160
130 CP=CP-100
140 N3=N3+1
150 GOTO 120
160 IF CP <= 50 GOTO 200
170 CP=CP-50
180 N4=N4+1
190 GOTO 160
200 IF CP <= 20 GOTO 240
210 CP=CP-20
220 N5=N5+1
230 GOTO 200
240 IF CP <= 10 GOTO 280
250 CP=CP-10
260 N6=N6+1
270 GOTO 240
280 IF CP <= 5 GOTO 320
290 CP=CP-5
300 N7=N7+1
310 GOTO 280
320 IF CP  < 1 GOTO 20
330 CP=CP-1
340 N8=N8+1
350 GOTO 320
360 LPRINTN1;N2;N3;N4;N5;N6;N7;N8
370 END


10000
0
0  1  4  1  2  0  1  5


10000
1574
567
0
1  3  4  3  3  1  2  11
```

OBSERVACIONES AL PROGRAMA

A) LOGICA

-El cálculo de las denominaciones 500,50,10 y 5 no requieren regresar al IF, ya que que a lo sumo se utilizará un solo billete (o moneda) de estos.

-Al calcular las monedas de 1 no se requiere IF, ya que CP: está en ese momento entre 0 y 5.

-La lógica es bastante clara.

B) PROGRAMACION

-El programa utiliza bytes.

-En BASIC R.S. se usa menos memoria si se agrupan varias instrucciones en una misma línea (hasta 255 bytes)

C) UTILIDAD DEL PROGRAMA

-Sería conveniente que se pudiera conocer el desglose de cada cantidad además del total.

-Mensajes en los PRINT ayudan a identificar resultados.

-Los resultados a la impresora ayudan a utilizarlos posteriormente.

CONCLUSION

-Desarrollar otro programa con las modificaciones necesarias para mejorarlo en los tres aspectos anteriores.

```
10 'DESGLOSE DE CANTIDADES SEGUNDA VERSION CARL FEB 80
20 V1=1000:V2=500:V3=100:V4=50:V5=20:V6=10:V7=5:V8=40:C=0:U=1:D=2
30 INPUT CP:IF CP<=C THEN GOTO 210  ELSELPRINT "LA CANTIDAD";CP;"SE DESGLOSA EN"
40 N1=C:N2=C:N3=C:N4=C:N5=C:N6=C:N7=C:T=T+CP
50 IF CP<= V1THEN GOTO 60   ELSE CP=CP-V1:N1=N1+U:GO TO 50
60 IF CP>V2 THEN CP=CP-V2:N2=U
70 IF CP<=V3 THEN GOTO80   ELSE CP=CP-V3:N3=N3+U:GOTO70
80 IF CP>V4 THEN CP=CP-V4:N4=U
90 IF CP>V8 THEN CP=CP-V8:N5=D ELSE IF CP>V5 THEN CP=CP-V5:N5=U
100 IF CP<=V6 THEN GOTO 110  ELSE CP=CP-V6:N6=N6+U:GOTO100
110 IF CP>V7 THEN CP=CP-V7:N7=U
120 IF N1>C THEN LPRINT N1;"DE MIL"
130 IF N2>C THEN LPRINT N2;"DE QUINIENTOS"
140 IF N3>C THEN LPRINT N3;"DE CIEN"
150 IF N4>C THEN LPRINT N4;"DE CINCUENTA"
160 IF N5>C THEN LPRINT N5;"DE VEINTE"
170 IF N6>C THEN LPRINT N6;"DE DIEZ"
180 IF N7>C THEN LPRINT N7;"DE CINCO"
190 IF CP>C THEN LPRINT CP;"DE UNO"
200 M1=M1+N1:M2=M2+N2:M3=M3+N3:M4=M4+N4:M5=M5+N5:M6=M6+N6:M7=M7+N7:M8=M8+CP:GOTO 30
210 LPRINT:LPRINT:LPRINT"T O T A L E S":IF M1>C THEN LPRINT M1;"DE MIL"
220 IF M2>C THEN LPRINT M2;"DE QUINIENTOS"
230 IF M3>C THEN LPRINT M3;"DE CIEN"
240 IF M4>C THEN LPRINT M4;"DE CINCUENTA"
250 IF M5>C THEN LPRINT M5;"DE VEINTE"
260 IF M6>C THEN LPRINT M6;"DE DIEZ"
270 IF M7>C THEN LPRINT M7;"DE CINCO"
280 IF M8>C THEN LPRINT M8;"DE UNO"
290 LPRINT "SUMA TOTAL",T
300 END
```

```
LA CANTIDAD 1000 SE DESGLOSA EN
  1 DE QUINIENTOS
  4 DE CIEN
  1 DE CINCUENTA
  2 DE VEINTE
  1 DE CINCO
  5 DE UNO

LA CANTIDAD 1574 SE DESGLOSA EN
  1 DE MIL
  1 DE QUINIENTOS
  1 DE CINCUENTA
  1 DE VEINTE
  4 DE UNO

LA CANTIDAD 567 SE DESGLOSA EN
  2 DE QUINIENTOS
  1 DE CINCUENTA
  1 DE DIEZ
  1 DE CINCO
  2 DE UNO

T O T A L E S
  1 DE MIL
  3 DE QUINIENTOS
  4 DE CIEN
  3 DE CINCUENTA
  3 DE VEINTE
  1 DE DIEZ
  2 DE CINCO
 11 DE UNO
SUMA TOTAL      3141
```

TECNICAS PARA EL DESARROLLO DE PROGRAMAS

ANTECEDENTES

-Grandes avances en la velocidad, capacidad y economía del Hardware.

-Surgimiento de nuevos y mejores lenguajes de programación.

-Creciente complejidad de las aplicaciones automatizadas en las empresas.

-Presupuesto dedicado al desarrollo y mantenimiento de sistemas E.D.P.

-Actual importancia de los sistemas E.D.P. en las empresas.

-Falta de un método matemático para demostrar la validez de un programa.

CONSECUENCIAS

Búsqueda y surgimiento de técnicas formales para facilitar el desarrollo de sistemas E.D.P.:

1) Programación estructurada
2) Pseudocódigo
3) Segmentación
4) Desarrollo descendente
5) HIPØ
6) Bibliotecas de soporte

Una característica fundamental de todas las técnicas anteriores es la sencillez de conceptos que involucran de la cual derivan su gran aceptación y correspondiente éxito.

PROGRAMA CONVENCIONAL

```
        IF p GOTO label q
        IF w GOTO label m
        L function
        GOTO label k
label m N function
        GOTO label k
label q IF q GOTO label t
        A function
        B function
        C function
label r IF NOT r GOTO label s
        D function
        GOTO label r
label s IF s GOTO label f
        E function
label v IF NOT v GOTO label k
        J function
label k K function
        END function
lable f F function
        GOTO label v
label t IF t GOTO label s
        A function
        B function
        GOTO label w
label s A function
        B function
        G function
label u IF NOT u GOTO label w
        H function
        GOTO label u
label w IF NOT t GOTO label y
        I function
label y IF NOT v GOTO label k
        J function
        GOTO label k
```

PROGRAMA ESTRUCTURADO

```
①IF p THEN
    A function
    B function
  ②IF q THEN
    ③IF t THEN
        C function
      ④DOWHILE u
        H function
      ④ENDDO
        I function
    ③(ELSE)
    ③ENDIF
  ②ELSE
      C function
    ③DOWHILE r
      D function
    ③ENDDO
    ③IF s THEN
        F function
    ③ELSE
        E function
    ③ENDIF
  ②ENDIF
  ②IF v THEN
      J function
  ②(ELSE)
  ②ENDIF
①ELSE
  ②IF w THEN
    M function
  ②ELSE
      L function
  ②ENDIF
①ENDIF
  K function
  END function
```

Figure 1. A comparison of structured and unstructured code


```
1Ø ' W=1:Q=1:S=1:B=-1:T=-1:U=1:V=1:P=1
2Ø 'W=Q=S=B=T=U=V=P=1
3Ø IF P GOTO9Ø
4Ø IF W GOTO7Ø
5Ø GOSUB 39Ø
6Ø GOTO2ØØ
7Ø GOSUB 4ØØ
8Ø GOTO2ØØ
9Ø IF Q GOTO24Ø
1ØØ GOSUB 41Ø
11Ø GOSUB 43Ø
12Ø IF NOT R GOTO 16Ø
13Ø GOSUB 44Ø
14Ø GOTO 12Ø
15Ø IF S GOTO 22Ø
16Ø GOSUB45Ø
17Ø IF NOT V GOTO 2ØØ
18Ø GOSUB46Ø
19Ø GOSUB47Ø
2ØØ END
21Ø GOSUB48Ø
22Ø GOTO 16Ø
23Ø IF T GOTO26Ø
24Ø GOSUB41Ø
25Ø GOSUB42Ø
26Ø GOTO34Ø
27Ø GOSUB41Ø
28Ø GOSUB42Ø
29Ø GOSUB49Ø
3ØØ IF NOT U GOTO 34Ø
31Ø GOSUB5ØØ
32Ø GOTO31Ø
33Ø IF NOT T GOTO 36Ø
34Ø GOSUB51Ø
35Ø IF NOT V GOTO 2ØØ
36Ø GOTO 2ØØ
37Ø END
38Ø PRINT"L":RETURN
39Ø PRINT"M":RETURN
4ØØ PRINT"A":RETURN
41Ø PRINT"B":RETURN
42Ø PRINT"C":RETURN
43Ø PRINT"D":RETURN
44Ø PRINT"E":RETURN
45Ø PRINT"J":RETURN
46Ø PRINT"K":RETURN
47Ø PRINT"F":RETURN
48Ø PRINT"G":RETURN
49Ø PRINT"H":RETURN
5ØØ PRINT"I":RETURN
51Ø PRINT"J":RETURN
52Ø PRINT"J":RETURN
```

U.N.A.M.	HOJA DE DISEÑO-DOCUMENTACION	H I P O	C E C A F I
SISTEMA:		AUTOR:	HOJA DE
SUBSISTEMA:		REFERENCIA:	FECHA:
APLICACION:		FUNCION:	TEOREMA DE LA ESTRUCTURA

ENTRADA

Cualquier problema de programación

PROCESO

Reglas del Teorema de la Estructura (1) (Programación Estructurada).

1.- Usar solamente las siguientes estructuras lógicas (estructuras básicas).

- Secuencia de una o más operaciones. (SEQUENCE).
- Bifurcación a una de dos operaciones y regreso. (IF THEN ELSE).
- Repetición de una operación mientras una condición sea verdadera. (DO WHILE).

2.- Combinar las estructuras:

- Sustituyendo una por otra
- Anidando una dentro de otra

(1) Bohm, C. and Jacopini, G. "Flow Diagrams, Turing Machines and Languages With only two formation Rules" Com. A.C.M. 9, No. 3 (Mayo 1966), 366-371.

SALIDA

Un programa propio o correcto. (Una entrada y una salida).

```
1 W=1:O=1:S=1:V=1:B=1:M=1:U=1:P=1:
2
IF P THEN
    GOSUB 1000:
    GOSUB 1010:
    GOSUB 1015:
ELSE
    IF W THEN
        GOSUB 1180:
    ELSE
        GOSUB 1190:
    ENDIF
ENDIF
20 GOSUB 1200 :END
1000 PRINT"A":RETURN
1010 PRINT"B":RETURN
1015
IF O THEN
    GOSUB 1020
ELSE
    GOSUB 1090
1016 RETURN
1020
IF T THEN
    GOSUB 1040:
    GOSUB 1050:
    GOSUB 1060:
1030 RETURN
1040 PRINT"G":RETURN
1050
IF O THEN
    GOSUB 1070:
    GOTO 1050:
1060 RETURN
1070 PRINT"H":O=0:RETURN
1080 PRINT"I":RETURN
1090
GOSUB 1110:
GOSUB 1120:
IF S THEN
    GOSUB"1150
ELSE
    GOSUB 1160
1100 RETURN
1110 PRINT"C":RETURN
1120
IF R THEN
    GOSUB 1140:
    GOTO 1120:
1130 RETURN
1140 PRINT"D":R=0:RETURN
1150 PRINT"F":RETURN
1160 PRINT"E":RETURN
1170 PRINT"J":RETURN
1180 PRINT"M":RETURN
1190 PRINT"L":RETURN
1200 PRINT"K":RETURN
```

CARACTERISTICAS DEL PSEUDO CODIGO

- INDEPENDIENTE DEL LENGUAJE
  MENOS MODIFICACIONES
- OBLIGA A PROGRAMAR EN FORMA ESTRUCTURADA
  SOLO CONTIENE ESTRUCTURAS BASICAS
- SUSTITUYE AL DIAGRAMA DE BLOQUE Y AL DIAGRAMA DE FLUJO
  ELIMINA TIEMPO DE DIBUJO
- FACIL DE ENTENDER
  NUESTRO IDIOMA
- SE APLICA A CUALQUIER NIVEL
  PROGRAMAS ó SISTEMAS
- MUESTRA NIVELES DE LOGICA
- FACILMENTE MODIFICABLE
  TRABAJO SECRETARIAL
- SIMPLIFICA LA CODIFICACION EN UN LENGUAJE ESPECIFICO
  CONVENCIONES DE CODIFICACION

PRACTICAS ASOCIADAS A LA PROGRAMACION ESTRUCTURADA

- SANGRIA DE LOGICAS DEPENDIENTES
  AYUDA A IDENTIFICAR LAS ESTRUCTURAS.
  MUESTRA EL NIVEL DE ANIDAMIENTO.
  ESTRUCTURAS AL MISMO NIVEL DE LOGICA SE
  COLOCAN AL MISMO NIVEL.
- LOGICAS COMPLETAS EN UNA PAGINA (PANTALLA)
  PERMITE LEER Y ENTENDER TODA UNA LOGICA
  SIN REFERENCIAS EXTERNAS.
  LAS SUBRUTINAS RESULTAN PARTICULARMENTE
  UTILES EN ESTOS CASOS.
- UTILIZAR COMENTARIOS
  HACE MAS EXPLICITA LA LOGICA.
  AYUDA A ENTENDER Y MODIFICAR LOS PROGRAMAS.
- NO ANIDAR MAS DE TRES NIVELES DE LOGICA SIMULTANEOS
  PUEDE CAUSAR CONFUSIONES
  PUEDE ALARGAR DEMASIADO UNA LOGICA COMPLETA
  (MAS DE UNA PAGINA).

E C U A C I O N

LEE A,B,C

DO WHILE HAYA DATOS

CALCULA DISCRIMINANTE

IF DISCRIMINANTE > 0 THEN

RAICES REALES DIFERENTES

ELSE:

IF DISCRIMINANTE = 0 THEN

RAICES REALES IGUALES

ELSE

RAICES COMPLEJAS

ENDIF

ENDIF

LEE A,B,C

ENDDO

CALCULA DISCRIMINANTE

DIS = $B^2 - 4 A C$

RAICES REALES DIFERENTES

. RA 1 = (- B + $\sqrt{DIS}$) /(2 A)

. RA 2 = (- B - $\sqrt{DIS}$) /(2 A)

. IMPRIME RA 1

. IMPRIME RA 2

RAICES REALES IGUALES

. RA 1 = - B/(2 A)

. RA 2 = R A 1

. IMPRIME R A 1

. IMPRIME R A 2

RAICES COMPLEJAS

. PARE 1 = - B/2A

. PARE 2 = PARE 1

. PAIM 1 = $\sqrt{-DIS}$ / 2 A

. PAIM 2 = PAIM 1

. IMPRIME PARE 1 " + " PAIM 1 " IMA "

. IMPRIME PARE 2 " - " PAIM 2 " IMA "

VECTORES

. Lee VECTOR UNO (al final poner HV)

. Lee VECTOR DOS (al final poner HV)

. Iniciar I, J

DOWHILE Elemento I ≠ IIV ó elemento J ≠ HV

DOWHILE Elemento I > elemento J

. Solo en 2

ENDDO

DOWHILE Elemento I > elemento J

. Solo en 1

ENDDO

DOWHILE Elemento I = elemento J y elemento I ≠ HV

En ambos

ENDDO

ENDDO

INICIAR I , J

. I ← 1

. J ← 1

SOLO EN 2

Escribe " El elemento solo está en el vector DOS "

J ← J + 1

SOLO EN 1

Escribe "El elemento solo está en el vecto UNO"

I ← I + 1

EN AMBOS

Escribe "El elemento esta en ambos vectores "

J ← J + 1

I ← I + 1

Q U I C K

R A D I O S H A C K

SISTEMA TRS-80 MODELO III

M I C R O C O M P U T A D O R A R A D I O S H A C K

S I S T E M A T R S - 8 0 M O D E L O I I I

INICIALIZACION.

El sistema completo (la computadora y los periféricos) debe estar apagado.

1. Primeramente encender todos los periféricos y después la computadora.

2. El mensaje: Cass?
debe aparecer en la pantalla. Para seleccionar una alta velocidad de grabación en cassette (1500 bauds), presione la tecla H o ENTER . Para seleccionar una baja velocidad de grabación en cassete (500 bauds), presione la tecla L .

   Para propósitos generales, use una velocidad de grabación alta. Para salvar o cargar un programa en Basic Nivel II Modelo I, se debe usar la velocidad de grabación baja.

3. El mensaje: Memory Size?
debe aparecer en la pantalla. Para utilizar toda la memoria disponible, presione la tecla ENTER . Para reservar algo de espacio en memoria, teclee la dirección más alta (en decimal) que usted quiera usar, entonces presione la tecla ENTER .

4. El mensaje:
```
Model III Basic
(c) Tandy'80
READY
>
```
debe aparecer en la pantalla. Entonces la computadora está lista para utilizarse.

INSTRUCCIONES.

- AUTO inicio, incremento
  El número de línea aparecerá automáticamente.
  AUTO AUTO 150 ,20 AUTO ,5

- CLEARn
  Asigna n bytes para cada variable tipo string; inicializa todas las variables.
  CLEAR CLEAR 75 CLEAR 0

- CLOAD
  Para cargar en la memoria de la computadora un programa en Basic salvado en cassette. Si se especifica un nombre sólo se tomará en cuenta la primera letra.
  CLOAD CLOAD "PRUEBA"

- CLOAD?
  Compara un programa en cinta magnética, byte por byte, con el programa residente en la computadora.
  CLOAD? CLOAD? "PRUEBA"

- CLS
  Limpia la pantalla.
  CLS

- CONT
  Continúa la ejecución de un programa después de una interrupción por una instrucción STOP o por presionar la tecla BREAK .
  CONT

- CSAVE
  Salva un programa residente en cinta magnética. Se debe especificar un nombre. Sólo el primer caracter del nombre es usado.
  CSAVE "PRUEBA"

- DATA
  Guarda datos que serán llamados por una instrucción READ.
  DATA "HERNANDEZ,V.",1960

- DEFDBL
  Define variables de doble presición.
  DEFDBL V,X-Z

- DEFINT
  Define variables de tipo entero.
  DEFINT A,I-N

- DEFSNG
  Define variables de presición simple.
  DEFSNG I,W-Z

- DEFSTR
  Define variables de tipo string.
  ```
  DEFSTR C,L-Z
  ```

- DELETE
  Borra líneas de un programa.
  ```
  DELETE 1205    DELETE -80    DELETE 100-300
  ```

- DIM
  Dimensiona uno o más arreglos.
  ```
  DIM R(75),W(40)    DIM L%(3,18,5)    DIM AR$(8,25)
  ```

- EDIT
  Pone a la computadora en modo editor para la línea especificada. Vea comandos del modo editor.
  ```
  EDIT 100    EDIT.
  ```

- END
  Finaliza la ejecución del programa.
  ```
  END
  ```

- ERROR(n)
  Simula el error especificado entre paréntesis de acuerdo al código de mensajes de error ($1 \leq n \leq 23$).
  ```
  ERROR(1)
  ```

- FOR...TO...STEP/NEXT
  Expresión iterativa.
  ```
  FOR J=1 TO 8 (...) NEXT J
  FOR C!=0 TO 5 STEP 2 (...) NEXT C!
  ```

- GOSUB
  Transfiere el control a la subrutina indicada.
  ```
  GOSUB 750
  ```

- GOTO
  Transfiere el control a la línea indicada.
  ```
  GOTO 180
  ```

- IF...THEN...ELSE
  Expresión condicional.
  ```
  IF P=Q THEN 200
  IF N%<0 THEN 150 ELSE N%=N%-1
  ```

- INPUT
  Para lectura de datos en pantalla.
  ```
  INPUT X#    INPUT L,M,N    INPUT "DATO";N
  ```

- INPUT#-1
  Para lectura de datos desde cassette.
  ```
  INPUT#-1,NUM
  ```

- LET
  Asigna un valor a una variable (opcional).

  ```
  LET X=7.05     LET R2=R1     LET C$="ROJO"
  ```

- LIST
  Lista las líneas del programa en la pantalla.

  ```
  LIST     LIST 50-85
  ```

- LLIST
  Lista las líneas del programa en la impresora.

  ```
  LLIST     LLIST 50-
  ```

- LPRINT
  Imprime en la impresora letrero(s) y/o variable(s).

  ```
  LPRINT CAP$,"ES EL CAPITAL DE";ST$
  ```

- LPRINT TAB
  Mueve el carro de la impresora a la posición especificada.

  ```
  LPRINT TAB(25)"REPORTE"
  ```

- LPRINT USING
  Imprime con el formato especificado, numeros y strings en la impresora. Vea PRINT USING para opciones de esta instrucción.

  ```
  LPRINT USING "####,"; 1234
  ```

- NEW
  Borra el programa en memoria; inicializa todas las variables.

  ```
  NEW
  ```

- ON ERROR GOTO
  Si hay un error transfiere el control hasta la línea especificada.

  ```
  ON ERROR GOTO 2100
  ```

- ON ERROR GOTO 0
  Inutiliza la rutina de error.

  ```
  ON ERROR GOTO 0
  ```

- ON...GOSUB
  Transfiere el control a alguna de las subrutinas especificadas dependiendo del valor de la variable.

  ```
  ON Y GOSUB 50,100,150,200
  ```

- ON...GOTO
  Transfiere el control a alguna de las líneas especificadas dependiendo del valor de la variable.

  ```
  ON X GOTO 190,200,210
  ```

- OUTp, v
  Envía el valor especificado en v a la localidad p ($0 \leq p \leq 255$).

  ```
  OUT 255, 0
  ```

- POKE n, v
  Pone el valor v(0≤v≤255) en la localidad n (desde 15360 hasta el fin de la memoria). Vea direccionamiento del POKE.
  POKE 15872, 255

- PRINT
  Imprime letreros y/o variables en la pantalla.
  PRINT X!+Y! PRINT "MEXICO"

- PRINT↷n
  Imprime empezando en n(0≤n≤1023)
  PRINT↷477, "CENTRO"

- PRINT#-1
  Graba datos en cinta magnética.
  PRINT#-1, A

- PRINT TAB
  Mueve el cursor hacia la derecha hasta la posición indicada.
  PRINT TAB(20) "REPORTE"

- PRINT USING
  Formatos para números y string.

  # Formato para números.
  PRINT USING "#####"; 66.2

  . Punto decimal
  PRINT, USING "##.#"; 58.76

  , Aparecerá una coma cada tercer dígito.
  PRINT USING "####,"; 1234

  ** Llena los primeros espacios con asterisco.
  PRINT USING "**####"; 44.0

  $$ Signo flotante de pesos.
  PRINT USING "$$##.##"; 118.6735

  **$ Signo flotante de pesos; llena los primeros espacios con asteriscos.
  PRINT USING "**$#.##"; 8.333

  ↑ Formato para exponenciales. Presione la tecla ↑ para que aparezca este caracter.
  PRINT USING "###.#↑↑↑↑"; 8527100

  + En la primera posición provoca que el signo sea impreso; en la última posición provoca que el signo sea impreso después del número.
  PRINT USING "+###"; -216

- Imprime el signo menos después de un número negativo, si es un número positivo deja un espacio en blanco.
  PRINT USING "####.##-"; -8124.42

! Imprime el primer caracter del string.
  PRINT USING "!"; "HOLA"

%espacios%
Campo de un string; la longitud de campo es el número de espacios más dos.
  PRINT USING "% %"; "AZUL"

- RANDOM
Genera números en forma aleatoria.
  RANDOM

- READ
Toma valores de una instrucción DATA.
  READ T READ S$ READ NM$,EDAD

- REM
Comentario; instrucción que indica a la computadora que debe ignorar el resto de la línea. El apóstrofe (') es una abreviación de REM.
  REM ESTA LINEA ES UN COMENTARIO 'ESTA TAMBIEN

- RESET (x,y)
Apaga el punto especificado de la gráfica.
  x: eje horizontal ($0 \leq x \leq 127$). y: eje vertical ($0 \leq y \leq 47$).
  RESET (21, 40) RESET (L1, L2)

- RESTORE
Cuando una instrucción READ aparece después de una instrucción RESTORE, entonces se volverán a utilizar los mismos datos de la primera instrucción DATA.
  RESTORE

- RESUME
Termina la rutina de error y ejecuta la línea indicada.
  RESUME 40 RESUME RESUME NEXT

- RETURN
Transfiere el control a la siguiente instrucción después de un GOSUB.
  RETURN

- RUN
Ejecuta el programa residente o una porción de él.
  RUN RUN 500

- SET (x, y)
Prende el punto indicado.
  x: eje horizontal ($0 \leq x \leq 127$). y: eje vertical ($0 \leq y \leq 47$).
  SET (10,0) SET (L1, L2)

- STOP
  Detiene la ejecución de un programa.
  ```
  STOP
  ```

- SYSTEM
  Pone a la computadora en modo monitor para cargar en disco archivos en lenguaje de máquina. A lo cual responderá con un *?, entonces teclee el nombre del archivo o /dirección.
  ```
  SYSTEM
  ```

- TROFF
  Anula la función TRACE.
  ```
  TROFF
  ```

- TRON
  Prende la función TRACE.
  ```
  TRON
  ```

C O D I G O D E C O N T R O L D E V I D E O

DECIMAL	HEXADECIMAL	PRINT CHR$ (código)
8	08	Espacios en blanco y borra el caracter actual.
10	0A	Imprime una línea y regresa el cursor
13	0D	Imprime una línea y regresa el cursor.
14	0E	Aparece el cursor.
15	0F	Desaparece el cursor.
21	15	Switchea caracteres especiales y compressión.
22	16	Switchea alternativamente los caracteres.
23	17	Cambia a modo de 32 caracteres por línea.
24	18	Mueve el cursor a la derecha sin borrar.
25	19	Avanza el cursor.
26	1A	Baja una línea.
27	1B	Sube una línea.
28	1C	Coloca el cursor al inicio de la pantalla.
29	1D	Mueve el cursor al inicio de la línea.
30	1E	Borra toda la línea.
31	1F	Limpia al final de la pantalla.

O P E R A D O R E S

Cada operador o grupo de operadores tiene menor prioridad que el anterior en el orden siguiente:

| o [— Exponenciación (para precisión simple). Presione la tecla ↑ para que aparezca este operador; deberá aparecer el paréntesis cuadrado izquierdo.

\- , + — Negativo unario, positivo.

* , / — Multiplicación, división.

= , - — Suma y concatenación (string), substracción.

<, >, =, <=, >=, <> — Comparaciones.

NOT

AND

OR

C A R A C T E R E S E S P E C I A L E S

'	Abreviación de la instrucción REM.
%	Para variables enteras.
!	Para variables de precisión simple.
#	Para variables de doble precisión.
$	Para variables de tipo string.
:	Separa instrucciones en una misma línea.
?	Abreviación de la instrucción PRINT (no es válido L? en vez de LPRINT).
,	Puntuación en la instrucción PRINT; para impresión en zonas de 16 columnas.
;	Puntuación de la instrucción PRINT, para separar letreros y/o variables en la declaración de la instrucción, pero en la impresión no separa con espacios.

C O M A N D O S E N M O D O E D I T O R

A	Anula los cambios y empieza otra vez.
n C	Cambia n caracteres.
n D	Borra n caracteres.
E	Sale del modo EDIT y actualiza los cambios realizados.
H	Corta la línea e inserta al final.
I	Inserta caracteres.
n K c	Borra todos los caracteres después de la n-ésima ocurrencia de c.
L	Lista la línea.
Q	Se desocupa el modo editor y cancela todos los cambios.
n S c	Busca la ocurrencia de c, tantas veces el valor de n.
X	Inserta al final de la línea.
SHIFT ↑	Sale del modo editor.
ENTER	Reconoce los cambios, permaneciendo en modo editor.
n SPACERBAR	Mueve el cursor n espacios a la derecha.
n ←	Mueve el cursor n espacios a la izquierda.

T E C L A S D E C O N T R O L

←	Borra el último caracter tecleado, regresa el cursor un espacio.
SHIFT ←	Borra la línea.
BREAK	Interrumpe alguna actividad y regresa al nivel de comandos.
CLEAR	Limpia la pantalla.
ENTER	Comunica el fin de línea.
SPACEBAR	Pone un blanco.
→	Avanza el cursor a la siguiente posición.
SHIFT →	Pone en modo 32-caracteres.
↓	Brinca una línea y pone el cursor.
SHIFT ↓	Tecla de "control", Presione simultáneamente estas dos y alguna tecla A-Z para lograr control A hasta control Z.
SHIFT ↓ *	Copia el contenido de la pantalla a impresora.
SHIFT @	Estatifica la ejecución de un programa (pausa). Para continuar presione alguna tecla.

F U N C I O N E S

Los argumentos estan indicados con letras especiales (variables):

REALES
x: (-1x10E38, -1x10E-38), (1x10E-38, 1x10E38)

ENTEROS
n: (-32768, 32767)

LOCALIDADES
c: (0, 255)

str: argumento tipo string.

var: nombre de variable.

- ABS(x)
Valor absoluto de x.
```
Y = ABS(x)
```

- ASC(str)
Proporciona el equivalente en código ASCII del primer caracter del string.
```
A = ASC(T$)
```

- ATN(x)
Arcotangente de x, en radianes.
```
Y = ATN(x/3)
```

- CDBL(x)
Convierte a la variable x en variable de doble precisión.
```
x# = CDBL(n*3)
```

- CHR$(c)
Proporciona el caracter equivalente en código ASCII, control o código de gráficas.
```
P$=CHR$(T)
```

- CINT(n)
Proporciona el entero más grande no mayor que n.
```
PRINT CINT(15.0075)
```

- COS(x)
Coseno del argumento dado en radianes.
```
Y = COS(x)
```

- CSNG(x)
Convierte a presición simple.
```
FC = CSNG(TH#)
```

- ERL
Proporciona el número de línea en la cual ha ocurrido un error.
```
PRINT ERL
```

- ERR
Si hay un error, proporciona el valor de acuerdo al código de error, este valor será (código de error .1) *2
```
IF ERR=12 THEN 650 ELSE 800
```

- EXP(x)
Calcula el antilogaritmo natural de x.
```
Y = EXP(x)
```

- FIX(x)
Proporciona todos los dígitos a la izquierda del punto.
```
Y = FIX(x)
```

- FRE(número)
Proporciona la cantidad de memoria disponible.
```
F=FRE(x)
```

- FRE(str)
Proporciona el espacio no usado por el string. Str es cualquier consonante o variable tipo string.
```
FRE("c")      FRE(c$)
```

- INKEY$
Proporciona al caracter tecleado, si está disponible.
```
A$ = INKEY$
```

- INP(p)
Proporciona el valor de la localidad p($0 \leq p \leq 255$).
```
V = INP(255)
```

- INT(x)
Proporciona el entero no mayor que x.
```
Y = INT(x)
```

- LEFT$(str,c)
Proporciona los primeros c caracteres de str.
```
P$ = LEFT$(M$,7)
```

- LEN(str)
Proporciona el número de caracteres del string.
```
X = LEN(SEN$)
```

- LOG(x)
Calcula el logaritmo natural de x.
```
Y = LOG(x)
```

- MEM
Proporciona la cantidad de memoria disponible.
```
PRINT MEM
```

- MID$(str, posición, longitud)
  Proporciona un substring de otro string. Si la longitud es omitida, el string a partir de posición será proporcionado.
  PRINT MID$(A$,2,3) F$=MID$(A$,3)

- PEEK(n)
  Da el valor guardado en la localidad n.
  V = PEEK(18520)

- POINT(x, y)
  Verifica si el punto de la gráfica está prendido o apagado.
  x: eje horizontal (0≤x≤127). y: eje vertical (0≤y≤47).
  IF POINT(13,35) THEN PRINT "PRENDIDO" ELSE PRINT "APAGADO"

- POS(x)
  Proporciona la columna de la posición del cursor (0-63).
  X es un argumento dummy.
  PRINT TAB(40) POS(0)

- RIGHT$(str, c)
  Proporciona la parte derecha del string a partir de c.
  ZIP$ = RIGHT$(AD$,5)

- RND(n)
  Genera un número aleatorio entre 1 y n, si n>1 o entre 0 y 1 si n=0.
  Y=RND(100) PRINT RND(0) R=RND(x)

- SGN(x)
  Proporciona el signo: -1, 0, 1 cuando x es negativo, cero o positivo.
  x = SGN(A*B)

- SIN(x)
  Calcula el seno donde el argumento debe estar en radianes.
  Y = SIN(x)

- SQR(x)
  Calcula la raíz cuadrada de x.
  Y = SQR(A+B)

- STR$(x)
  Convierte una expresión numérica en string.
  S$ = STR$(x)

- STRING$(l, c)
  Proporciona un string de longitud l. Dónde c puede ser en código ASCII o como un string.
  B$ = STRING$(125,"?") B$ = STRING$(125,63)

- TAN(x)
  Calcula la tangente cuyo argumento es en radianes.
  Y = TAN(x)

- TIME$
  Proporciona el tiempo (con formato de 24 horas) y el dato es un string de 17 caracteres.
  ```
  A$ = TIME$
  ```

- USR(x)
  Llama una subrutina en lenguaje de máquina localizada en la dirección x(16526-16527)
  ```
  PRINT USR(-1)     Y=USR(x)
  ```

- VAL(str)
  Convierte el string a un número.
  ```
  V% = VAL("100 PESOS")
  ```

- VARPTR(var)
  Proporciona la dirección donde la variable ha sido guardada.
  ```
  Y = USR(VARPTR(x))
  ```

M E N S A J E S D E E R R O R

CODIGO	ABREVIACION	EXPLICACION
1	NF	Next sin su correspondiente FOR.
2	SN	Error de sintaxis.
3	RG	Return sin su correspondiente GOSUB.
4	OD	Faltaron datos en una instrucción READ o INPUT.
5	FC	Llamada incorrecta de una función.
6	OV	Se excedió la capacidad de un número.
7	OM	Memoria saturada.
8	UL	Linea indefinida.
9	BS	Subíndice fuera de rango.
10	DD	Arreglo dimensionado nuevamente.
11	/0	División por cero.
12	ID	Se utiliza la instrucción INPUT como comando directo.
13	TM	Conflicto de operandos.
14	OS	El espacio total de un string fue excedido.
15	LS	String con muchos caracteres.
16	ST	Fórmula del string muy complicada.
17	CN	No puede continuar.
18	NR	No hay una instrucción RESUME.
19	RW	Instrucción RESUME SIN ERROR.
20	UE	Error indefinido.
21	MO	Error de operandos.
22	FD	Archivo de datos mal salvado.
23	L3	Solo para el sistema de Basic con disco.

DIRECCIONAMIENTOS PARA LA INSTRUCCION P O K E

Las siguientes localidades pueden activar o controlar algunas de las características especiales de la TRS-80 Modelo III. Vea el Manual de Operación para mayores detalles.

Por ejemplo:
Para seleccionar una alta velocidad de grabación en cassette, ejecute: POKE 16913,1

LOCALIDAD DEC	LOCALIDAD HEX	CONTENIDO	CONTENIDO INICIAL
16409	4019	Switch de letras n=0: Letras mayúsculas y minúsculas n≠0: Sólo letras mayúsculas	Mayúsculas
16412	401C	Parpadeo del cursor n=0: Parpadea n≠0: No parpadea	0
16416	4020	Localidad del cursor Dos bytes: LSB, MSB	N/A
16419	4023	Caracter del cursor Código ASCII (0-255)	176
16424	4028	Máximo de líneas/página más uno	67
16425	4029	Número de líneas impresas más uno	1
16427	402B	Longitud máxima de línea menos dos	255
16526	408E	Localidad de la rutina USR Dos bytes: LSB, MSB	7754
16872	41E8	Buffer de lectura $RSRCV Un byte	0
16880	41F0	Buffer de salida $RSTX Un byte	0
16888	41F8	Velocidad de transferencia de código	85
16889	41F9	Código de paridad. Longitud de palabra y Stop-bit	108
16890	41FA	Switch de espera $RSINIT n=0: No espera n≠0: Espera	Espera
16913	4211	Switch de la velocidad de grabación en cassette n=0: 500 bauds n≠0: 1500 bauds	N/A

LOCALIDAD DEC	LOCALIDAD HEX	CONTENIDO	CONTENIDO INICIAL
16916	4214	Protección de líneas en pantalla, desde 0 a 7. Para valores mayores son interpretados en módulo 8.	0
16928	4220	Dispositivo $ROUTE Dos bytes de E/S designados.	N/A
16930	4222	Dispositivo de default $ROUTE Dos bytes de E/S designados.	N/A

S U B R U T I N A S D E M E M O R I A R O M Z - 8 0

Las siguientes subrutinas ROM pueden ser usadas por un programa en Z-80; algunas de ellas pueden ser usadas en programas de Basic por medio de la función USR. Antes de usar alguna de éstas, lea la sección de información técnica del manual de operación.

LOCALIDAD DEC	LOCALIDAD HEX	CONTENIDO	FUNCION
0	0000	$RESET	Apaga el sistema.
43	002B	$KBCHAR	Checa el caracter presionado.
51	0033	$VDCHAR	Aparece un caracter.
59	003B	$PRCHAR	Imprime un caracter.
64	0040	$KBLINE	Espera un comando.
73	0049	$KBWAIT	Espera un caracter.
80	0050	$RSRCV	Recibe un caracter desde RS-232-C.
85	0055	$RSTX	Transmite un caracter a RS-232-C.
90	005A	$RSINIT	Inicializa el RS-232-C.
96	0060	$DELAY	Define por un tiempo especificado.
105	0069	$INITIO	Inicializa todos los manejadores de E/S.
108	006C	$ROUTE	Ruta de E/S.
457	01C9	$VDCLS	Limpia la pantalla.
473	01D9	$PRSCN	Imprime el contenido de la pantalla.
539	021B	$VDLINE	Aparece una línea.
565	0235	$CSIN	Lectura de un byte en cassette.
612	0264	$CSOUT	Salida de un byte en cassette.

LOCALIDAD DEC	LOCALIDAD HEX	CONTENIDO	FUNCION
647	0287	$CSHWR	Escribe la cabeza del cassette.
653	0280	$KBBRK	Checa la tecla BREAK.
662	0296	$CSHIN	Lee la cabeza del cassette.
664	0298	$CLKON	Aparece el reloj en la pantalla.
673	02A1	$CKLOFF	Desaparece el reloj de la pantalla.
6681	1A19	$READY	Pasa a Basic, apareciendo "READY".
12339	3033	$DATE	Da la fecha.
12342	3036	$TIME	Da el tiempo.
12354	3042	$SETCAS	Prende la grabadora.
12312	37E8	$PRSTAT	Estado normal de la impresora (realiza lecturas), solo cuando: Bit 7=0 "Está desocupada". Bit 6=0 "Hay papel". Bit 5=1 "Dispositivo seleccionado". Bit 4=1 "La impresora no está saturada". Los bits 3, 2, 1 y 0 no se usan.

PROBLEMA I) DESGLOSE DE CANTIDADES

ENUNCIADO

Se desea el número de billetes y monedas necesarias para pagar una lista de raya. Se conocen las cantidades a pagar, todas múltiplos de un peso.

SOLUCION

Análisis del problema:

Se utilizarán billetes de 1000, 500, 100, 50 y 20 pesos y monedas de 10, 5 y 1 pesos.

Se utilizará una denominación solo cuando el monto a desglosar lo exceda.

EJEMPLO:

$50 se descompone en $20+$20+$5+$1+$1+$1+$1+$1

```
1Ø 'DESGLOSE DE CANTIDADES      CARL.  FEB 8Ø
2Ø INPUT CP
25 LPRINTCP
3Ø IF CP <= Ø GOTO 36Ø
4Ø IF CP <= 1ØØØ GOTO 8Ø
5Ø CP=CP-1ØØØ
6Ø N1=N1+1
7Ø GOTO 4Ø
8Ø IF CP <= 5ØØ GOTO 12Ø
9Ø CP=CP-5ØØ
1ØØ N2=N2+1
11Ø GOTO 8Ø
12Ø IF CP <=1ØØ GOTO 16Ø
13Ø CP=CP-1ØØ
14Ø N3=N3+1
15Ø GOTO 12Ø
16Ø IF CP <= 5Ø GOTO 2ØØ
17Ø CP=CP-5Ø
18Ø N4=N4+1
19Ø GOTO 16Ø
2ØØ IF CP <= 2Ø GOTO 24Ø
21Ø CP=CP-2Ø
22Ø N5=N5+1
23Ø GOTO 2ØØ
24Ø IF CP <= 1Ø GOTO 28Ø
25Ø CP=CP-1Ø
26Ø N6=N6+1
27Ø GOTO 24Ø
28Ø IF CP <= 5 GOTO 32Ø
29Ø CP=CP-5
3ØØ N7=N7+1
31Ø GOTO 28Ø
32Ø IF CP < 1 GOTO 2Ø
33Ø CP=CP-1
34Ø N8=N8+1
35Ø GOTO 32Ø
36Ø LPRINTN1;N2;N3;N4;N5;N6;N7;N8
37Ø END
```

```
1ØØØ
Ø
Ø  1  4  1  2  Ø  1  5
```

```
1ØØØ
1574
567
Ø
1  3  4  3  3  1  2  11
```

OBSERVACIONES AL PROGRAMA

A) LOGICA

-El cálculo de las denominaciones 500,50,10 y 5 no requieren regresar al IF, ya que que a lo sumo se utilizará un solo billete (o moneda) de estos.

-Al calcular las monedas de 1 no se requiere IF, ya que CP está en ese momento entre 0 y 5.

-La lógica es bastante clara.

B) PROGRAMACION

-El programa utiliza bytes.

-En BASIC R.S. se usa menos memoria si se agrupan varias instrucciones en una misma línea (hasta 255 bytes)

-En BASIC R.S. se ahorra memoria si se utilizan variables en vez de constantes.

C) UTILIDAD DEL PROGRAMA

-Sería conveniente que se pudiera conocer el desglose de cada cantidad además del total.

-Mensajes en los PRINT ayudan a identificar resultados.

-Los resultados a la impresora ayudan a utilizarlos posteriormente.

CONCLUSION

-Desarrollar otro programa con las modificaciones necesarias para mejorarlo en los tres aspectos anteriores.

```
1Ø 'DESGLOSE DE CANTIDADES SEGUNDA VERSION CARL FEB 8Ø
2Ø V1=1ØØØ:V2=5ØØ:V3=1ØØ:V4=5Ø:V5=2Ø:V6=1Ø:V7=5:V8=4Ø:C=Ø:U=1:D=2
3Ø INPUT CP:IF CP<=C THEN GOTO 21Ø  ELSELPRINT "LA CANTIDAD";CP;"SE DESGLOSA EN"
4Ø N1=C:N2=C:N3=C:N4=C:N5=C:N6=C:N7=C:T=T+CP
5Ø IF CP<= V1THEN GOTO 6Ø   ELSE CP=CP-V1:N1=N1+U:GO TO 5Ø
6Ø IF CP>V2 THEN CP=CP-V2:N2=U
7Ø IF CP<=V3 THEN GOTO8Ø   ELSE CP=CP-V3:N3=N3+U:GOTO7Ø
8Ø IF CP>V4 THEN CP=CP-V4:N4=U
9Ø IF CP>V8 THEN CP=CP-V8:N5=D ELSE IF CP>V5 THEN CP=CP-V5:N5=U
1ØØ IF CP<=V6 THEN GOTO 11Ø  ELSE CP=CP-V6:N6=N6+U:GOTO1ØØ
11Ø IF CP>V7 THEN CP=CP-V7:N7=U
12Ø IF N1>C THEN LPRINT N1;"DE MIL"
13Ø IF N2>C THEN LPRINT N2;"DE QUINIENTOS"
14Ø IF N3>C THEN LPRINT N3;"DE CIEN"
15Ø IF N4>C THEN LPRINT N4;"DE CINCUENTA"
16Ø IF N5>C THEN LPRINT N5;"DE VEINTE"
17Ø IF N6>C THEN LPRINT N6;"DE DIEZ"
18Ø IF N7>C THEN LPRINT N7;"DE CINCO"
19Ø IF CP>C THEN LPRINT CP;"DE UNO"
2ØØ M1=M1+N1:M2=M2+N2:M3=M3+N3:M4=M4+N4:M5=M5+N5:M6=M6+N6:M7=M7+N7:M8=M8+CP:GOTO 3Ø
21Ø LPRINT:LPRINT:LPRINT"T O T A L E S":IF M1>C THEN LPRINT M1;"DE MIL"
22Ø IF M2>C THEN LPRINT M2;"DE QUINIENTOS"
23Ø IF M3>C THEN LPRINT M3;"DE CIEN"
24Ø IF M4>C THEN LPRINT M4;"DE CINCUENTA"
25Ø IF M5>C THEN LPRINT M5;"DE VEINTE"
26Ø IF M6>C THEN LPRINT M6;"DE DIEZ"
27Ø IF M7>C THEN LPRINT M7;"DE CINCO"
28Ø IF M8>C THEN LPRINT M8;"DE UNO"
29Ø LPRINT "SUMA TOTAL",T
3ØØ END
```

```
LA CANTIDAD 1ØØØ SE DESGLOSA EN
 1 DE QUINIENTOS
 4 DE CIEN
 1 DE CINCUENTA
 2 DE VEINTE
 1 DE CINCO
 5 DE UNO


LA CANTIDAD 1574 SE DESGLOSA EN
 1 DE MIL
 1 DE QUINIENTOS
 1 DE CINCUENTA
 1 DE VEINTE
 4 DE UNO


LA CANTIDAD 567 SE DESGLOSA EN
 1 DE QUINIENTOS
 1 DE CINCUENTA
 1 DE DIEZ
 1 DE CINCO
 2 DE UNO


T O T A L E S
 1 DE MIL
 3 DE QUINIENTOS
 4 DE CIEN
 3 DE CINCUENTA
 3 DE VEINTE
 1 DE DIEZ
 2 DE CINCO
 11 DE UNO
SUMA TOTAL     3141
```

TECNICAS PARA EL DESARROLLO DE PROGRAMAS

ANTECEDENTES

-Grandes avances en la velocidad, capacidad y economía del Hardware.

-Surgimiento de nuevos y mejores lenguajes de programación.

-Creciente complejidad de las aplicaciones automatizadas en las empresas.

-Presupuesto dedicado al desarrollo y mantenimiento de sistemas E.D.P.

-Actual importancia de los sistemas E.D.P. en las empresas.

-Falta de un método matemático para demostrar la validez de un programa.

CONSECUENCIAS

Búsqueda y surgimiento de técnicas formales para facilitar el desarrollo de sistemas E.D.P.:

1) Programación estructurada

2) Pseudocódigo

3) Segmentación

4) Desarrollo descendente

5) HIPØ

6) Bibliotecas de soporte

Una característica fundamental de todas las técnicas anteriores es la sencillez de conceptos que involucran de la cual derivan su gran aceptación y correspondiente éxito.

PROGRAMA CONVENCIONAL

```
        IF p GOTO label q
        IF w GOTO label m
        L function
        GOTO label k
label m M function
        GOTO label k
label q IF q GOTO label t
        A function
        B function
        C function
label r IF NOT r GOTO label s
        D function
        GOTO label r
label s IF s GOTO label f
        E function
label v IF NOT v GOTO label k
        J function
label k K function
        END function
lable f F function
        GOTO label v
label t IF t GOTO label a
        A function
        B function
        GOTO label w
label a A function
        B function
        G function
label u IF NOT u GOTO label w
        H function
        GOTO label u
label w IF NOT t GOTO label y
        I function
label y IF NOT v GOTO label k
        J function
        GOTO label k
```

PROGRAMA ESTRUCTURADO

```
①IF p THEN
    A function
    B function
  ②IF q THEN
    ③IF t THEN
        G function
      ④DOWHILE u
          H function
      ④ENDDO
          I function
    ③(ELSE)
    ③ENDIF
  ②ELSE
      C function
    ③DOWHILE r
        D function
    ③ENDDO
    ③IF s THEN
        F function
    ③ELSE
        E function
    ③ENDIF
  ②ENDIF
  ②IF v THEN
      J function
  ②(ELSE)
  ②ENDIF
①ELSE
  ②IF w THEN
    M function
  ②ELSE
      L function
  ②ENDIF
①ENDIF
  K function
  END function
```

Figure 1. A comparison of structured and unstructured code

```
10 ' W=1:Q=1:S=1:B=-1:T=-1:U=1::V=1:P=1
20 'W=Q=S=B=T=U=V=P=1
30 IF P GOTO90
40 IF W GOTO 70
50 GOSUB 390
60 GOTO200
70 GOSUB 400
80 GOTO200
90 IF Q GOTO240
100 GOSUB 410
120 GOSUB 430
130 IF NOT R GOTO 160
140 GOSUB 440
150 GOTO 130
160 IF S GOTO 220
170 GOSUB450
180 IF NOT V GOTO 200
190 GOSUB460
200 GOSUB470
210 END
220 GOSUB480
230 GOTO 180
240 IF T GOTO280
250 GOSUB410
260 GOSUB420
270 GOTO340
280 GOSUB410
290 GOSUB420
300 GOSUB490
310 IF NOT U GOTO 340
320 GOSUB500
330 GOTO310
340 IF NOT T GOTO 360
350 GOSUB510
360 IF NOT V GOTO 200
370 GOTO 200
380 END
390 PRINT"L":RETURN
400 PRINT"M":RETURN
410 PRINT"A":RETURN
420 PRINT"B":RETURN
430 PRINT"C":RETURN
440 PRINT"D":RETURN
450 PRINT"E":RETURN
460 PRINT"J":RETURN
470 PRINT"K":RETURN
480 PRINT"F":RETURN
490 PRINT"G":RETURN
500 PRINT"H":RETURN
510 PRINT"I":RETURN
520 PRINT"J":RETURN
```

```
4 W=1:Q=1:S=1:V=1:B=1:T=1:U=1:P=1:
1Ø
IF P THEN
      GOSUB 1ØØØ:
      GOSUB 1Ø1Ø:
      GOSUB 1Ø15:
ELSE
      IF W THEN
         GOSUB 118Ø:
      ELSE
         GOSUB 119Ø:
 ENDIF
2Ø GOSUB 12ØØ :END
1ØØØ PRINT"A":RETURN
1Ø1Ø PRINT"B":RETURN
1Ø15
IF Q THEN
   GOSUB 1Ø2Ø
ELSE
   GOSUB 1Ø9Ø
1Ø16 RETURN
1Ø2Ø
IF T THEN
   GOSUB 1Ø4Ø:
   GOSUB 1Ø5Ø:
   GOSUB 1Ø8Ø:
1Ø3Ø RETURN
1Ø4Ø PRINT"G":RETURN
1Ø5Ø
IF U THEN
   GOSUB 1Ø7Ø:
   GOTO 1Ø5Ø:
1Ø6Ø RETURN
1Ø7Ø PRINT"H":U=Ø:RETURN
1Ø8Ø PRINT"I":RETURN
1Ø9Ø
GOSUB 111Ø:
GOSUB 112Ø:
IF S THEN
   GOSUB 115Ø
ELSE
   GOSUB 116Ø
11ØØ RETURN
111Ø PRINT"C":RETURN
112Ø
IF R THEN
   GOSUB 114Ø:
   GOTO 112Ø:
113Ø RETURN
114Ø PRINT"D":R=Ø:RETURN
115Ø PRINT"F":RETURN
116Ø PRINT"E":RETURN
117Ø PRINT"J":RETURN
118Ø PRINT"M":RETURN
119Ø PRINT"L":RETURN
12ØØ PRINT"K":RETURN
```

U.N.A.M. 143	HOJA DE DISEÑO-DOCUMENTACION TIPO	C E C A F I

SISTEMA:		AUTOR:	DE
SUBSISTEMA:		REFERENCIA:	FECHA:
APLICACION:		FUNCION: TEOREMA DE LA ESTRUCTURA	

ENTRADA

Cualquier problema de programación

PROCESO

Reglas del Teorema de la Estructura (1) (Programación Estructurada).

1.- Usar solamente las siguientes estructuras lógicas (estructuras básicas).

- Secuencia de una o más operaciones. (SEQUENCE).
- Bifurcación a una de dos operaciones y regreso. (IF THEN ELSE).
- Repetición de una operación mientras una condición sea verdadera. (DO WHILE).

2.- Combinar las estructuras:

- Substituyendo una por otra
- Anidando una dentro de otra

(1) Bohm, C. and Jacopini, G. "Flow Diagrams, Turing Machines and Languages With only two formation Rules" Com. A.C.M. 9, No. 3 (Mayo 1966), 366-371.

SALIDA

Un programa propio o correcto. (Una entrada y una salida).

CARACTERISTICAS DE UN PROGRAMA PROPIO

- SIEMPRE PUEDE OBTENERSE
  TEOREMA DE EXISTENCIA

- UNA ENTRADA Y UNA SALIDA
  PERMITE USARSE COMO OTRA ESTRUCTURA BASICA

- FACILIDAD DE LECTURA
  DE ARRIBA HACIA ABAJO

- NUNCA ROMPE SU SECUENCIA
  NO HAY BIFURCACIONES INCONDICIONALES (GØ TØ)

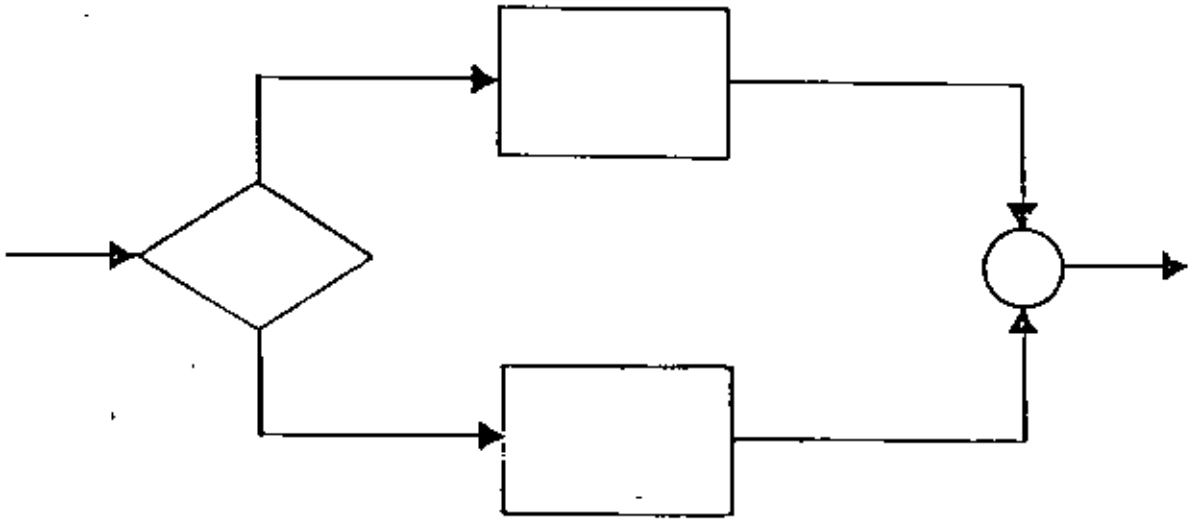
- FACIL DE PROBAR
  SU LOGICA ES EVIDENTE

- FACIL DE ENTENDER
  SOLO CONTIENE ESTRUCTURAS BASICAS

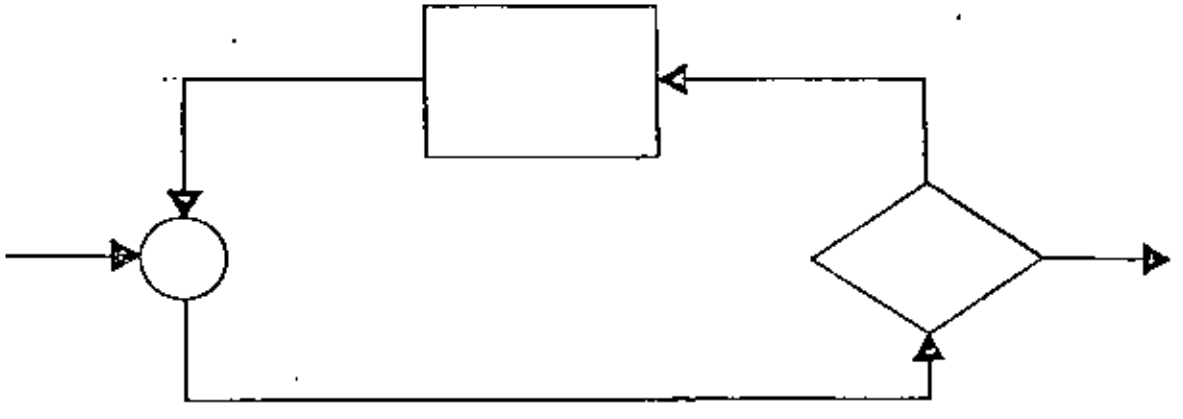
- FACILIDAD DE MANTENIMIENTO
  LOGICAS COMPLETAS EN UNA SOLA REGION

ESTRUCTURAS LOGICAS BASICAS

SEQUENCE

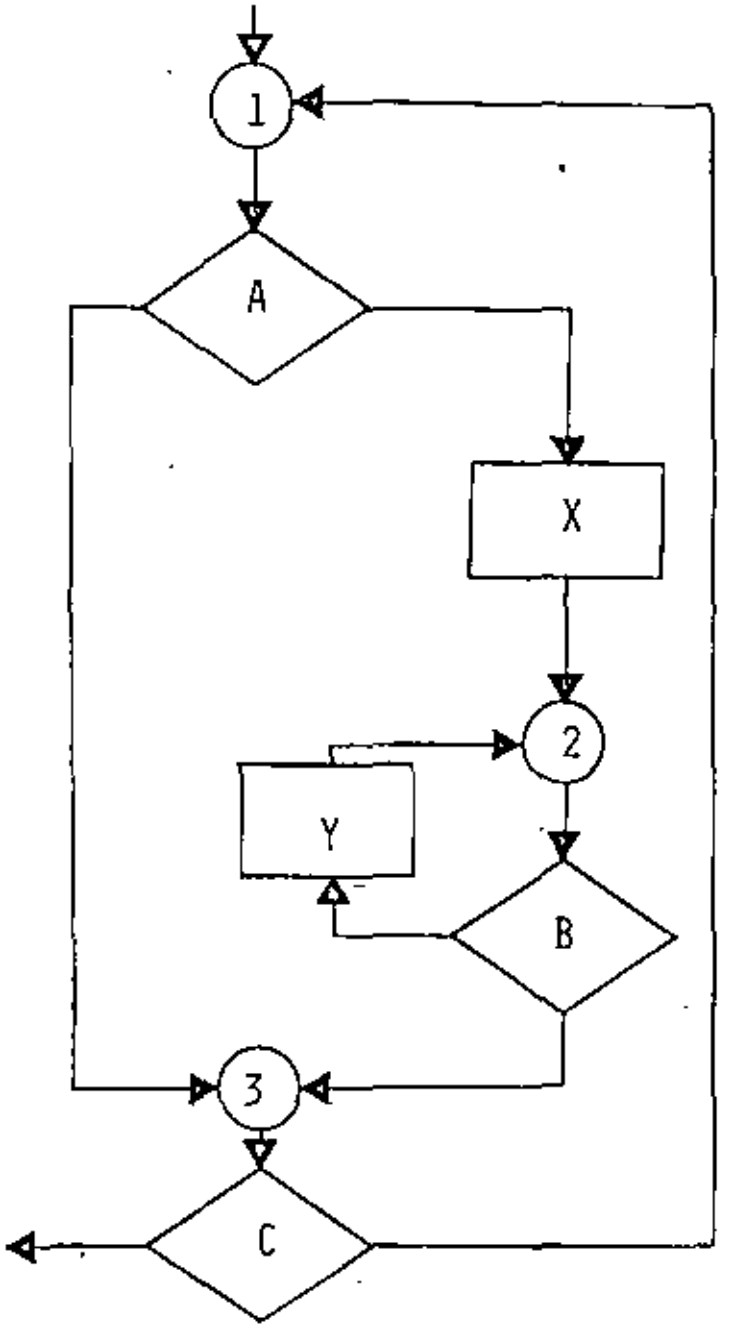


IFTHENELSE

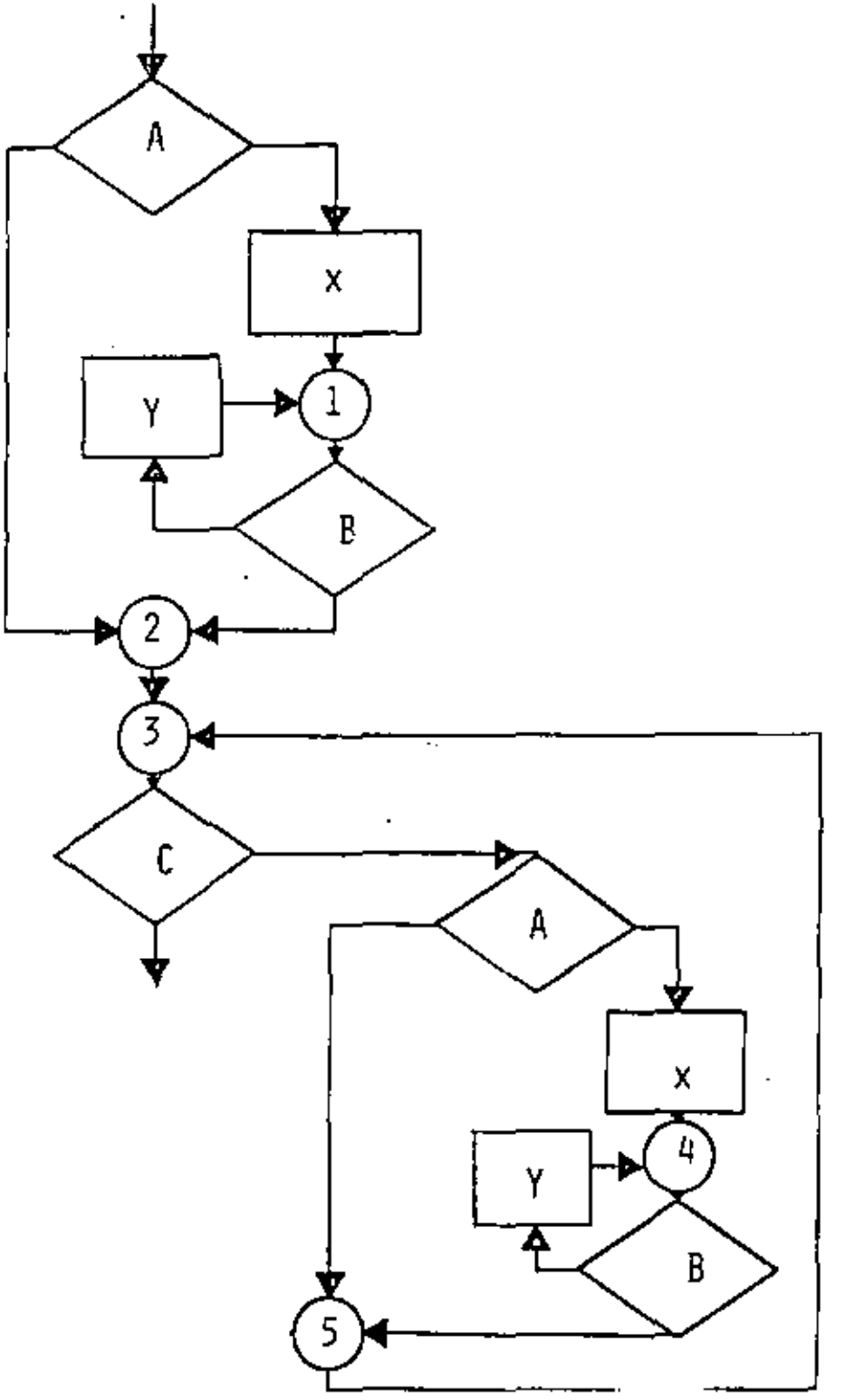


DOWHILE

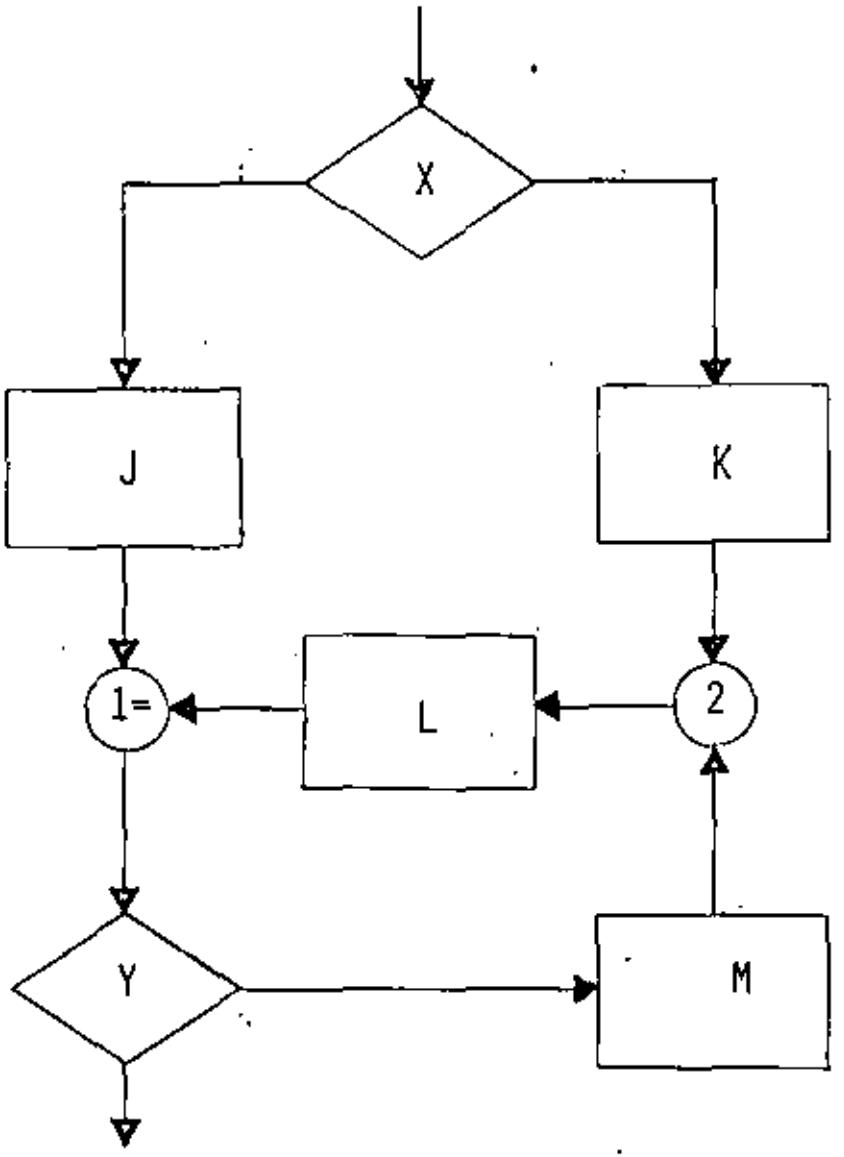
PROBLEMA



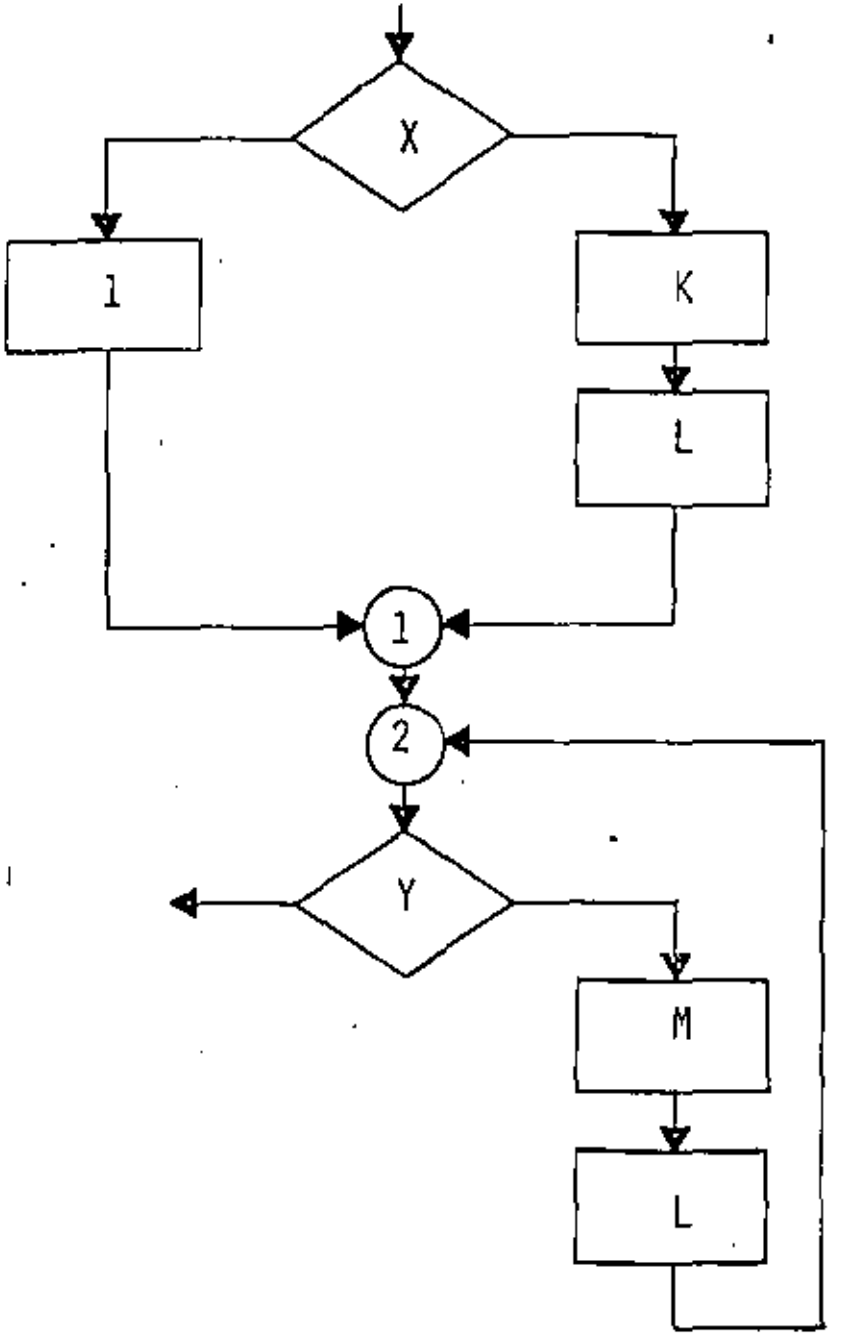
SØLUCION

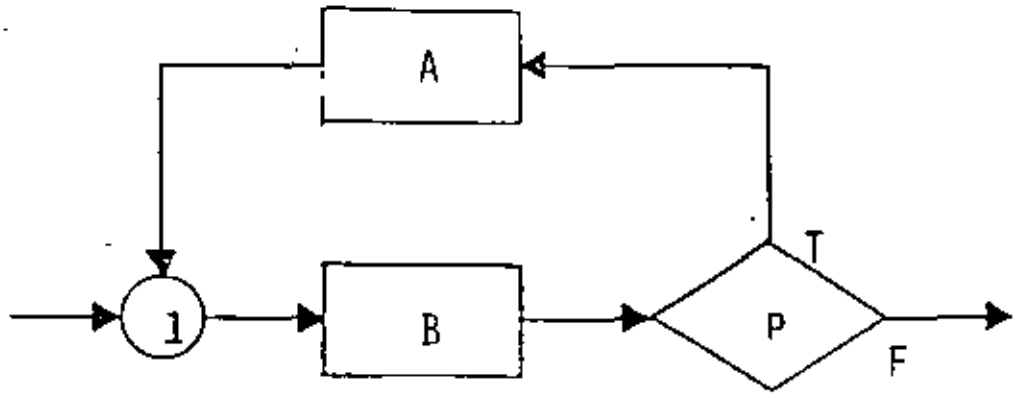


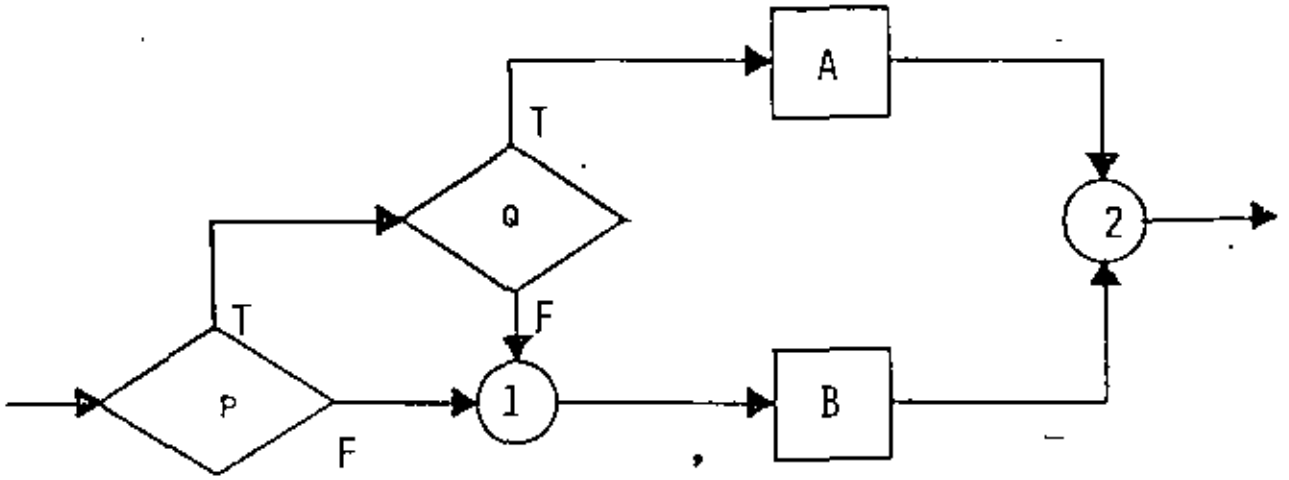
PROBLEMA

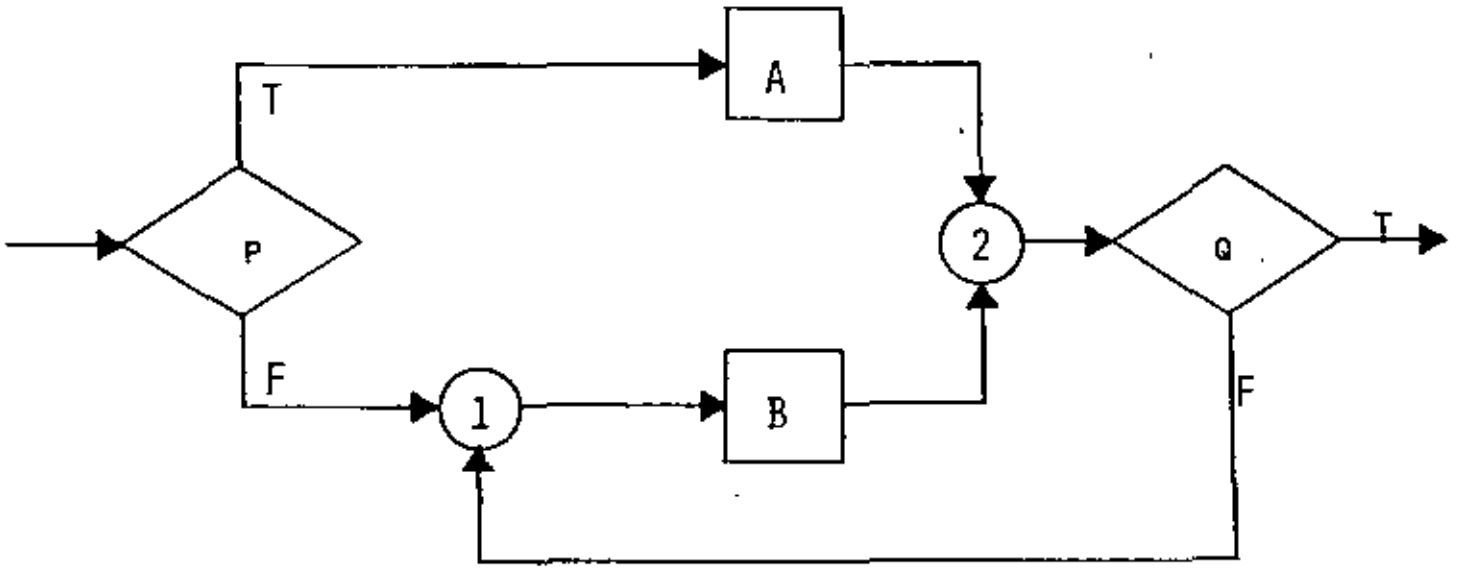


SØLUCION



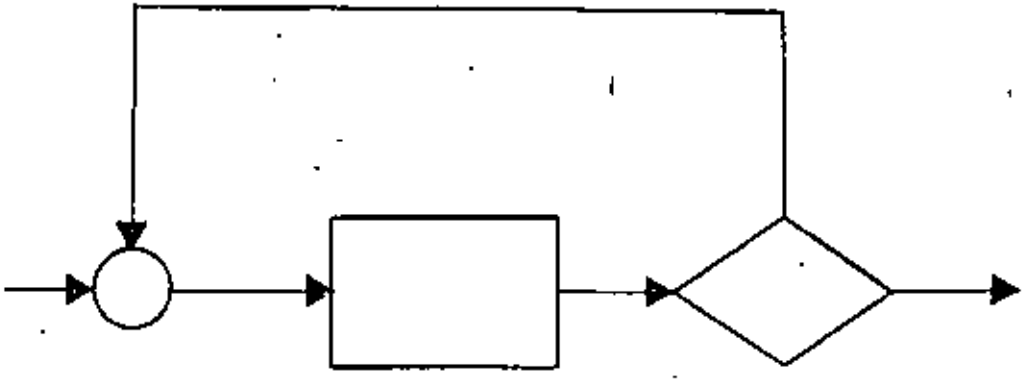

A
1
B
P
T
F


A
T
Q
2
T
F
P
1
B
F

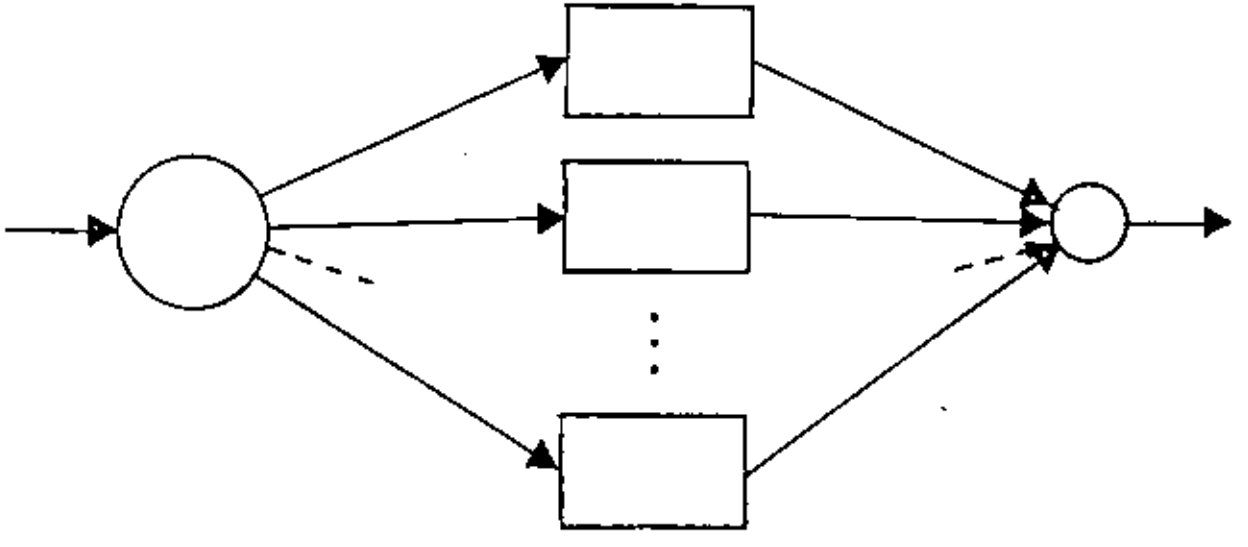

A
T
P
2
Q
T
F
1
B
F

FIGURAS LOGICAS ADICIONALES

(UNA ENTRADA Y UNA SALIDA)



DOUNTIL



CASE

U.N.A.M. 150 HOJA DE DISEÑO-DOCUMENTACION H I P O C E C A F I

SISTEMA:	AUTOR:	A DE
SUBSISTEMA:	REFERENCIA:	FECHA:
APLICACION:	FUNCION: CONVENCIONES DE PSEUDO CODIGO	

E N T R A D A

Cualquier problema de programación.

Programación Estructurada.

P R O C E S O

Convenciones de pseudo código

1.- SEQUENCE. Describir la operación.

2.- IF THEN ELSE. Usar el formato:

```
IF Condición THEN
    describir operaciones.
ELSE
    describir operaciones.
ENDIF
```

3.- DO WHILE. Usar el formato:

```
DOWHILE condición
    describir operaciones.
ENDDO
```

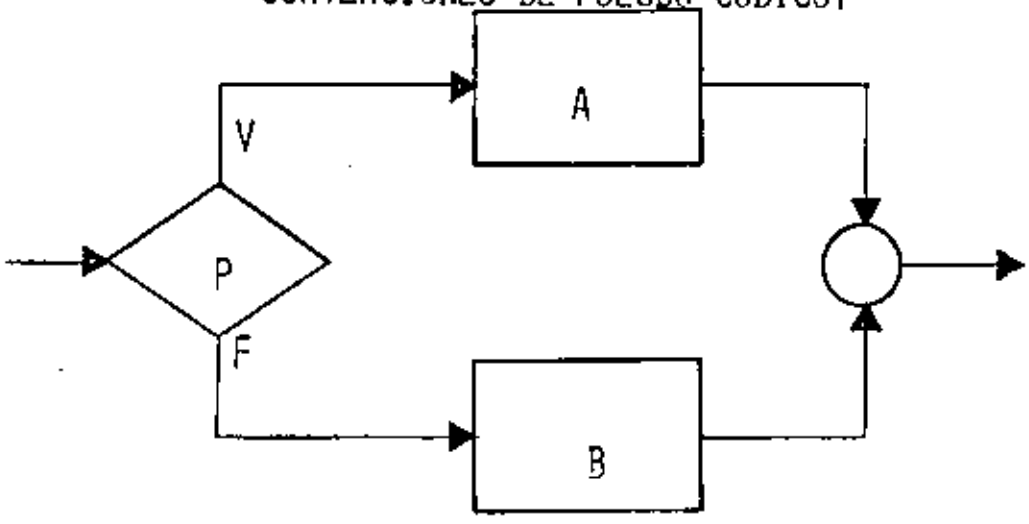
4.- Mediante un corrimiento a la derecha de tres o más posiciones (sangrado) se muestra la dependencia de una estructura respecto a otra.

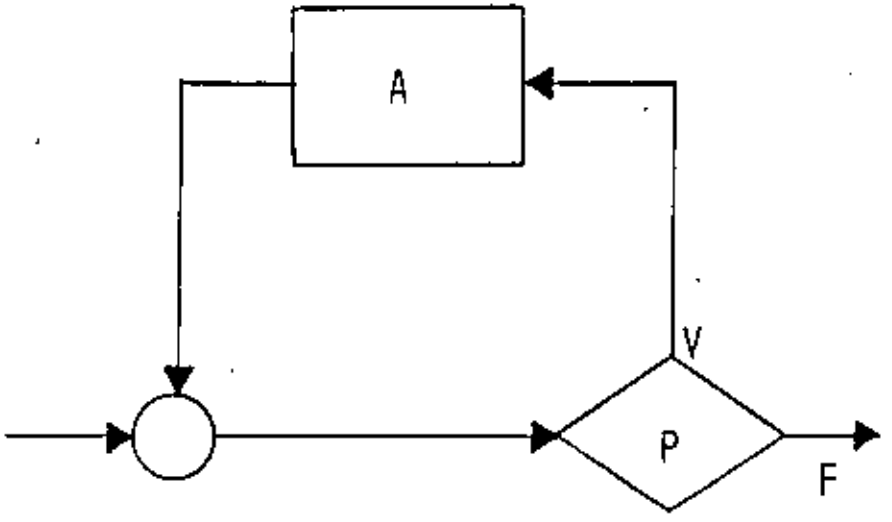
5.- Todas las operaciones y condiciones se describen usando todas las facilidades existentes en nuestro idioma.

S A L I D A

Narrativo usando todas las facilidades del idioma, de la lógica de un programa propio o correcto.

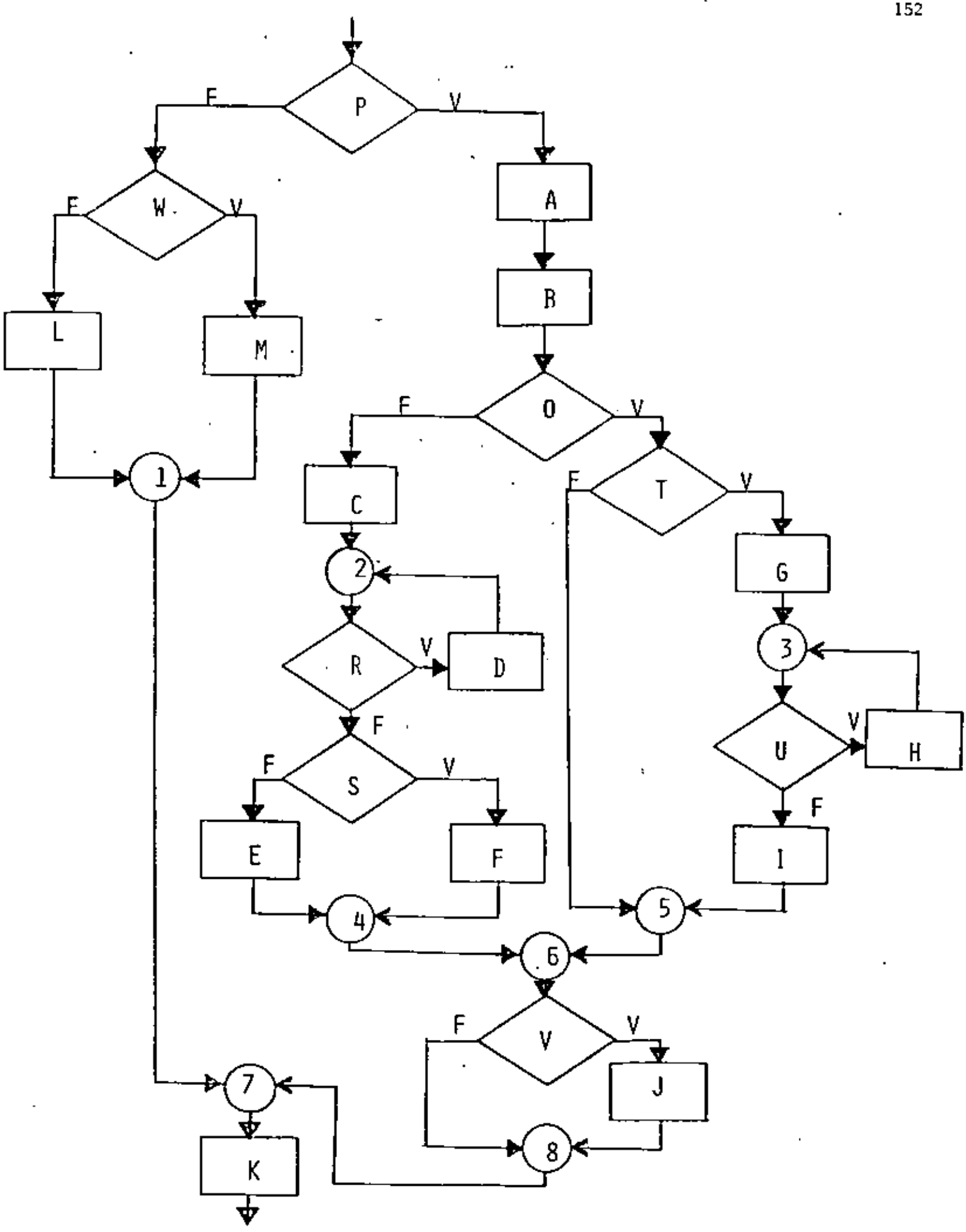
CONVENCIONES DE PSEUDO CODIGO.






```
IF     P     THEN
       FUNCION A
ELSE
       FUNCION B
ENDIF
```

```
DOWHILE     P
       FUNCION  A
ENDDO
```


P
F
V
W
F
V
L
M
A
B
O
F
V
C
T
F
V
G
2
R
V
D
3
U
V
H
F
S
F
V
E
F
I
F
4
5
6
V
F
V
J
7
8
K

CARACTERISTICAS DEL PSEUDO CODIGO

- INDEPENDIENTE DEL LENGUAJE
  - MENOS MODIFICACIONES
- OBLIGA A PROGRAMAR EN FORMA ESTRUCTURADA
  - SOLO CONTIENE ESTRUCTURAS BASICAS
- SUSTITUYE AL DIAGRAMA DE BLOQUE Y AL DIAGRAMA DE FLUJO
  - ELIMINA TIEMPO DE DIBUJO
- FACIL DE ENTENDER
  - NUESTRO IDIOMA
- SE APLICA A CUALQUIER NIVEL
  - PROGRAMAS 6 SISTEMAS
- MUESTRA NIVELES DE LOGICA
- FACILMENTE MODIFICABLE
  - TRABAJO SECRETARIAL
- SIMPLIFICA LA CODIFICACION EN UN LENGUAJE ESPECIFICO
  - CONVENCIONES DE CODIFICACION

PRACTICAS ASOCIADAS A LA PROGRAMACION ESTRUCTURADA

- SANGRIA DE LOGICAS DEPENDIENTES
  - AYUDA A IDENTIFICAR LAS ESTRUCTURAS.
  - MUESTRA EL NIVEL DE ANIDAMIENTO.
  - ESTRUCTURAS AL MISMO NIVEL DE LOGICA SE COLOCAN AL MISMO NIVEL.

- LOGICAS COMPLETAS EN UNA PAGINA (PANTALLA).
  - PERMITE LEER Y ENTENDER TODA UNA LOGICA SIN REFERENCIAS EXTERNAS.
  - LAS SUBRUTINAS RESULTAN PARTICULARMENTE UTILES EN ESTOS CASOS.

- UTILIZAR COMENTARIOS
  - HACE MAS EXPLICITA LA LOGICA.
  - AYUDA A ENTENDER Y MODIFICAR LOS PROGRAMAS.

- NO ANIDAR MAS DE TRES NIVELES DE LOGICA SIMULTANEOS
  - PUEDE CAUSAR CONFUSIONES
  - PUEDE ALARGAR DEMASIADO UNA LOGICA COMPLETA (MAS DE UNA PAGINA).

E C U A C I O N

```
LEE A,B,C
DO WHILE HAYA DATOS
   CALCULA DISCRIMINANTE
   IF DISCRIMINANTE > 0 THEN
     RAICES REALES DIFERENTES
   ELSE:
     IF DISCRIMINANTE = 0 THEN
       RAICES REALES IGUALES
     ELSE
       RAICES COMPLEJAS
     ENDIF
   ENDIF
   LEE A,B,C
ENDDO
```

CALCULA DISCRIMINANTE

$DIS = B^2 - 4\ A\ C$

RAICES REALES DIFERENTES

. RA 1 = (- B + $\sqrt{\text{DIS}}$) /(2 A)

. RA 2 = (- B - $\sqrt{\text{DIS}}$) /(2 A)

. IMPRIME RA 1

. IMPRIME RA 2

RAICES REALES IGUALES

. RA 1 = - B/(2 A)

. RA 2 = R A 1

. IMPRIME R A 1

. IMPRIME R A 2

RAICES COMPLEJAS

. PARE 1 = - B/2A

. PARE 2 = PARE 1

. PAIM 1 = $\sqrt{-\text{DIS}}$ / 2 A

. PAIM 2 = PAIM 1

. IMPRIME PARE 1 " " + " PAIM 1 " IMA "

. IMPRIME PARE 2 " - " PAIM 2 " IMA "

. Lee VECTOR UNO$_i$ (al final poner HV)

. Lee VECTOR DOS$_j$ (al final poner HV)

. Iniciar i, j

DOWHILE Elemento$_i$ ≠ HV ó elementos$_j$ ≠ HV

DOWHILE Elemento$_i$ > elemento$_j$

. Solo en 2

ENDDO

DOWHILE Elemento$_i$ > elemento$_j$

. Solo en 1

ENDDO

DOWHILE Elemento$_i$ = elemento$_j$ y
elemento$_i$ ≠ HV

En ambos

ENDDO

ENDDO

INICIAR i , j

. I ← 1

. J ← 1

SOLO EN 2

Escribe " El elemento solo está

en el vector DOS "

J ← J + 1

SOLO EN 1

Escribe "El elemento solo está en el vecto UNO"

I ← I + 1

EN AMBOS

Escribe "El elemento esta en ambos

vectores "

J ← J + 1

I ← I + 1

DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA U.N.A.M.

LENGUAJE DE PROGRAMACION BASIC CON APLICACIONES

(PRIMERA PARTE)

PROGRAMACION ESTRUCTURADA

FEBRERO, 1983

Palacio de Minería Calle de Tacuba 5 primer piso Deleg. Cuauhtemoc 06000 México, D.F. Tel.: 521-40-20 Apdo. Postal M-2285

OBSERVACIONES

1) CADA LINEA NUMERADA (LINEA FISICA) ADMITE HASTA 255 CARACTERES.

2) CADA LINEA FISICA CONSUME PER SE 5 LOCALIDADES DE MEMORIA POR LO QUE CONVIENE TRATAR DE UTILIZAR TODOS SUS CARACTERES.

3) COLOCAR UNA INSTRUCCION SEGUIDA INMEDIATAMENTE DE LAS OTRAS ES PERMITIDO (USO DE:), PERO OBSCURECE LA LOGICA, POR LO QUE CONVIENE COLOCAR CADA INSTRUCCION EN DIFERENTE LINEA (LINEA LOGICA) FORMANDO ASI UNA LINEA FISICA CON VARIAS LINEAS LOGICAS.

4) EL CAMBIO DE LINEA LOGICA SIN CAMBIAR DE LINEA FISICA PUEDE HACERSE CON LA TECLA ↓ (LINE FEED) LA CUAL CONSUME UN SOLO CARACATER.

5) PARA SANGRAR EL TEXTO CONVIENE UTILIZAR DOS O TRES ESPACIOS.

6) EL PROGRAMA RESULTANTE SIGUIENDO ESTAS PRACTICAS PUEDE SER UN POCO MAS GRANDE QUE UN PROGRAMA CONVENCIONAL, PERO SUS VENTAJAS SOBRE ESTE ULTIMO LO RECOMPENSAN MUCHAS MAS VECES.

C O N V E N C I O N E S D E C O D I F I C A C I O N

A) SEQUENCE (SECUENCIA)

```
10
A = B:
C = D:
READ E, F:
INPUT G, H:
.
.                          240 CARACTERES POR LINEA FISICA O
                           255 CON EDIT.
20
DATA 1, 2:
RESTORE
```

ETC.

INSTRUCCIONES SEQUENCE EN BASIC LEVEL II TRS 80

CLEAR, CLS, DATA, DEFDBL, DEFINT, DEFSNG, DEFSTR, DIM, END, ERROR, GOSUB, INPUT, LET, ONGOSUB, OUT, POKE, PRINT, RANDOM, READ, REM, RESTORE, RESET, RETURN, SET, STOP,

B) DO WHILE

```
30
IF A > B  THEN  70 :
35
   C = D :
   PRINTE :
   :
   :
   GØ TØ 30
'ENDDO          FIN DEL DØ
70
F = G
:
:
:
```

SE PERMITEN VARIAS LINEAS FISICAS
EN EL RANGO DEL DOWHILE

C) DOUNTIL

```
80
FOR I = 1   TØ N STEP 2 :
    J = I+1 :
    K = L*J :
    :
    :
NEXT  I
```

SE PERMITEN VARIAS LINEAS FISICAS
EN EL RANGO DEL DOUNTIL

D) IF THEN ELSE

```
20
IF  A < B THEN
    C = D+E:                    LA INSTRUCCION IF THEN ELSE DEBE
    F = G-H                     CABER TOTALMENTE EN UNA LINEA
ELSE                            (? SN ERROR).
    PRINT   I,J
'ENDIF                  FIN DEL IF
```


E) CASE

```
ON I GØ TØ 20, 30, 40
20  A = B*C :
    D = E :
    :
    GØ TØ 50
30  G = H :                     SE PERMITEN VARIAS LINEAS FISICAS
    :                           EN CADA RANGO
    :
    GØ TØ 50
40  I=J ↑ K :
    :
    GØ TØ 50                    PUEDE OMITIRSE
50
L = M+N
:
:
:
```

F). GØ TØ

MANEJAR CØN EXTREMØ CUIDADØ

. GØ TØ

. ØN ERRØR GØ TØ Y

. RESUME

LAS CUALES SØN FIGURAS DE TRANSFERENCIA

INCØNDICIØNAL Y CAUSAS DE MUCHOS PRØBLEMAS.

LA FIGURA GØ TØ NUNCA ES NECESARIA.

LAS INSTRUCCIONES ØN ERRØR GØ TØ Y RESUME PUEDEN SER

DE MUCHA AYUDA SI SE UTILIZAN PARA MANEJAR LAS CONDICIONES

DE ERRØR EXCLUSIVAMENTE.

C O M B I N A C I O N D E E S T R U C T U R A S B A S I C A S

A) DØ DENTRO DE IF

```
10
IF   A<>B  THEN
     FØR J=I TØ  N:
        PRINT J:
     NEXT J
ELSE
     PRINT A
ENDIF
20
L = M:
.
.
.
```

RECORDAR: IF THEN ELSE DEBE ESTAR TOTALMENTE INCLUIDO EN UNA LINEA FISICA

B) IF DENTRØ DE DØ

```
30
FØR K= 1 TØ 5 STEP 0,1:
    IF K=B  THEN
      PRINT K :
      :
    ELSE
      A= K*2:
      PRINT A:
NEXT  K
```

AL MENOS EL IF THEN ELSE DEBE CABER EN UNA SOLA LINEA FISICA. EL RANGO DEL FØR PUEDE TENER VARIAS LINEAS FISICAS

NØ PUEDE LLEVAR 'ENDIF SI EL NEXT ESTA EN LA MISMA LINEA FISICA (AL CØLOCAR REM ó ' , EL RESTO DE LA LINEA FISICA SE CØNSIDERA CØMENTARIØ)

C) IF DENTRØ DE IF

```
20
IF  A  THEN
    IF B  THEN
      D= I:
      C= D*2
    ELSE
      C=E
ELSE
      C=F:
      R=T:
      W=Z
'ENDIF
```

TØDO EL TEXTO DEBE CABER EN UNA LINEA FISICA (IF THEN ELSE). EN EL CASO QUE EL IF INTERNØ NØ TENGA ELSE, DEBERA INCLUIRSE EL ELSE INTERNØ CØN UNA INSTRUCCION MUDA CØMØ A=A PARA FØRZAR QUE EL ELSE DE LA CØLUMNA UNØ CORRESPONDA AL IF EXTERNØ.
EL IF INTERNO NØ PUEDE LLEVAR 'ENDIF (VER CASØ ANTERIØR)

170

```
10 'DESGLOSE DE CANTIDADES TRACMA VERSION  CRL FEB. 80
20 GOSUB 120 'VALORES INICIALES A LAS VARIABLES
30 INPUT CP
40
50 IF CP <=0 THEN 190
60    LPRINT "LA CANTIDAD", CP," SE DESGLOSA EN :"
      N1=C:N2=C:N3=C:N4=C:N5=C:N6=C:N7=C:
         T=T+CP
70    'DESGLOSE EN 100(GOSUB130),500(160),100(160),5(310)
                  20(230),10(250) Y 5(270)
80    GOSUB 130:
      GOSUB 160:
      GOSUB 180:
      GOSUB 210:
      GOSUB 230:
      GOSUB 250:
      GOSUB 270:
90 GOSUB 290 'ACUMULA PARCIALES EN TOTALES
95 GOSUB 300:
   INPUT CP:
   GOTO 40
100 GOSUB 390 'IMPRIME TOTALES FINALES
110 END

120
V1=100:V2=50:V3=10:V4=5:V5=2:V6=1:V7=5:V8=0:
C=0:U=1:D=2:
RETURN

130
IF CP<-V1 THEN 150
140 C=CP-V1:
    N1=N1+U:
    GOTO 130:
'ENDIF
150 RETURN
```

169

D) USO DE GOSUB EN IF QUE REQUERIRIAN MAS DE UNA LINEA FISICA

```
10
IF  A THEN
    GOSUB200          SIEMPPE CABE EN UNA SOLA LINEA FISICA
ELSE
    GOSUB300
'ENDIF
20
.
.
.
200
IF  B  THEN
    D= I:             ESTE IF DEBE CABER EN UNA SOLA LINEA, SI ESTO
    C= D*2            NO FUERA TODAVIA POSIBLE, SE REPETIRIA EL USO DEL GOSUB
ELSE                  (QUIZA GOSUB 400 Y GOSUB 500) A ESTE NIVEL Y EN LOS
    C=E:              QUE LE SIGAN DE SER NECESARIO
'ENDIF
210 RETURN            DEBE SER OTRA LINEA FISICA (UN IF QUE EJECUTA UN
300                   THEN AL TERMINAR SALTA A LA SIGUIENTE LINEA)

C= F:
R= T:
M= Z:
RETURN                PUEDE SER LA MISMA LINEA FISICA
```

```
16Ø
IF CP≥V2 THEN
   CP=CP-V2:
   N2=U
'ENDIF
17Ø RETURN
18Ø
IF CP<=V3 THEN 2ØØ
19Ø CP=CP-V3:
    N3=N3+U:
    GOTO 18Ø:
'ENDDO
2ØØ RETURN
21Ø
IF CP≥ V4 THEN.
   CP=CP-V4:
   N4=U
'ENDIF
22Ø RETURN
23Ø
IF CP≥V8 THEN
   CP=CP-V8:
   N5=0
ELSE
    IF CP≥V5 THEN
       CP=CP-V5:
       N5=U
    'ENDIF
ENDIF
24Ø RETURN
25Ø
IF CP≥V6 THEN
   CP=CP-V6:
   N6=U
'ENDIF
26Ø RETURN
27Ø
IF CP≥V7 THEN
   CP=CP-V7:
   N7=U
'ENDIF
28Ø RETURN

29Ø M1=M1+N1:M2=M2+N2:M3=M3+N3:M4=M4+N4:M5=M5+N5:M6=M6+N6:
M7=M7+N7:M8=M8+CP:
RETURN

3ØØ IF>N1  Ø THEN LPRINTN1;"DE $1ØØØ"
31Ø IF>N2  Ø THEN LPRINTN2;"DE $5ØØ"
32Ø IF>N3  Ø THEN LPRINTN3;"DE $1ØØ"
33Ø IF>N4  Ø THEN LPRINTN4;"DE $5Ø"
34Ø IF>N5  Ø THEN LPRINTN5;"DE $2Ø"
35Ø IF>N6  Ø THEN LPRINTN6;"DE $1Ø"
36Ø IF>N7  Ø THEN LPRINTN7;"DE $5"
37Ø IF>CP  Ø THEN LPRINTCP;"DE $1"
38Ø RETURN

39Ø
N1=N1:N2=M2:N3=M3:N4=M4:N5=M5:N6=M6:M7=M7:CP=M8:
LPRINT"TOTALES":
GOSUB 3ØØ:
LPRINT "SUMA TOTAL",T
RETURN
```

```
LA CANTIDAD 1ØØØ  SE DESGLOSA EN :
 1 DE $5ØØ
 4 DE $1ØØ
 1 DE $5Ø
 2 DE $2Ø
 1 DE $5
 5 DE $1


LA CANTIDAD 1574  SE DESGLOSA EN :
 1 DE $1ØØØ
 1 DE $5ØØ
 1 DE $5Ø
 1 DE $2Ø
 4 DE $1


LA CANTIDAD 567  SE DESGLOSA EN :
 1 DE $5ØØ
 1 DE $5Ø
 1 DE $1Ø
 1 DE $5
 2 DE $1


TOTALES
 1 DE $1ØØØ
 3 DE $5ØØ
 4 DE $1ØØ
 3 DE $5Ø
 3 DE $2Ø
 1 DE $1Ø
 2 DE $5
 11 DE $1
SUMA TOTAL        $3141
```

```
1Ø CLS
2Ø 'ESTE PROGRAMA ORDENA UN VECTOR DE MENOR A MAYOR
3Ø DIM A(2Ø)
4Ø INPUT"TAMAÑO DEL VECTOR",TV:
FOR I =1 TO TV:
   INPUT A(I):
NEXT I:
LPRINT "EL VECTOR ORIGINAL ES:":
FOR I=1 TO TV:
   LPRINT A(I):
NEXT I
5Ø
FOR I =1 TO TV-1:
   FOR J=I+1 TO TV:
      IF A(I)>A(J) THEN
         AUX=A(I):
         A(I)=A(J):
         A(J)=AUX:
      'ENDIF
6Ø NEXT J:
NEXT I:
LPRINT "EL VECTOR ORDENADO ES":
FOR I=1 TO TV:
   LPRINT A(I):
NEXT I
7Ø GOTO 4Ø
```

```
EL VECTOR ORIGINAL ES:
657
12.56
8975
25
4
EL VECTOR ORDENADO ES
4
12.56
25
657
8975
```

```
10 CLS
20 'ESTE PROGRAMA REALIZA UN CRUCE DE 2 ARCHIVOS ENTRE DOS
   VECTORES PREVIAMENTE ORDENADOS EN FORMA ASCENDENTE:
30 DIM A(20),B(20)
40 INPUT "TAMAÑO DEL VECTOR A",TA:
FOR I=1 TO TA:
   INPUT A(I):
NEXT I:
INPUT "TAMAÑO DEL VECTOR B",TB:
FOR I=1 TO TB:
   INPUT B(I):
NEXT I
50 CLS:LPRINT "EL VECTOR A ES":
FOR I=1 TO TA:
   LPRINT A(I),:
NEXT:
LPRINT" ":LPRINT "EL VECTOR B ES:":
FOR I=1 TO TB:
   LPRINT B(I),:
NEXT:LPRINT" ":LPRINT"LA CLASIFICACION DE LOS ELEMENTOS ES:":
A(TA+1)=199999:B(TB+1)=199999:
I=1:J=1:
60
IF A(I)=199999 AND B(J)=199999 THEN 140
70   IF A(I)<=B(J) THEN 90
80      GOSUB 150  :
        GOTO 70   :
    'ENDDO
90   IF A(I)>=B(J) THEN 110
100      GOSUB 160 :
         GOTO 90   :
    'ENDDO
110   IF A(I)  B(J) OR A(I)=199999 THEN 130
120      GOSUB 170  :
         GOTO 110  :
    'ENDDO
130   GOTO60
'ENDDO
140 GOTO 40
150 LPRINT B(J),"SOLO EN B":J=J+1:RETURN
160 LPRINT A(I),"SOLO EN A":I=I+1:RETURN
170 LPRINT A(I),"EN  AMBOS":I=I+1:J=J+1:RETURN
```

```
EL VECTOR A ES:
 2              3              4              5              6
EL VECTOR B ES :
 5              6              7              8              9
LA CLASIFICACION DE LOS ELEMENTOS ES:
 2              SOLO EN A
 3              SOLO EN A
 4              SOLO EN A
 5              EN  AMBOS
 6              EN  AMBOS
 7              SOLO EN B
 8              SOLO EN B
 9              SOLO EN B
```


EJEMPLO DE DOCUMENTACION

PROGRAMA: Desglose de cantidades

PROPOSITO: Desglose de un número cualquiera de cantidades en denominaciones de 1000,500,100,50,20,10, 5 y 1

AUTOR: Carlos A. Ramos L.

FECHA: Enero, 1980 .

PROTEGIDO EN: 1) Casete # 3 Q23-030 "p"

2) Disco # 7 "DESGLO/BAS"

DATOS NECESARIOS: Cantidades a desglosar

RESULTADOS: Cada cantidad leída se imprime junto con su desglose. Al final se escriben la suma de las cantidades leídas junto con el desglose total

METODO: Cada cantidad se va decrementando a medida que se van almacenando cantidades en las que se puede descomponer. Una denominación se utiliza solo cuando la cantidad restante o descomponer la excede.

EJEMPLO 50=20+20+5+1+1+1+1+1

RESTRICCIONES: Las cantidades deben ser multiplos de un peso (sin centavos)

USO DEL PROGRAMA:

Lenguaje: BASIC RADIO SHACK LEVEL II

Entrada de datos: Los datos se proporcionan por la pantalla. La última cantidad debe ser cero, la cual indica el fin de las cantidades.

EJEMPLO:

Entrada

? 1000

? 1574

? 567

? 0

Salida

(Ver hoja anterior)

BIBLIOGRAFIA:

OBSERVACIONES: Una copia del listado puede encontrarse en la carpeta 6 de la biblioteca.

FUNCIONAMIENTO INTERNO

Nombre de la variable	Significado
C	Constante de Valor 0
U	" 1
D	" 2
V1	" 1000
V2	" 500
V3	" 100
V4	" 50
V5	" 20
V6	" 10
V7	" 5
V8	" 40
CP	Cantidad a desglosar
T	Suma de cantidades
N1	Contador por cantidad de billetes de 1000
N2	" " " " " " 500
N3	" " " " " " 100
N4	" " " " " " 50
N5	" " " " " " 20
N6	" " " " monedas " 10
N7	" " " " " " 5
M1	Contador total de billetes de 1000
M2	" " " " 500
M3	" " " " 400
M4	" " " " 50
M5	" " " " 20
M6	" " " monedas 10
M7	" " " " 5
M8	" " " " 1

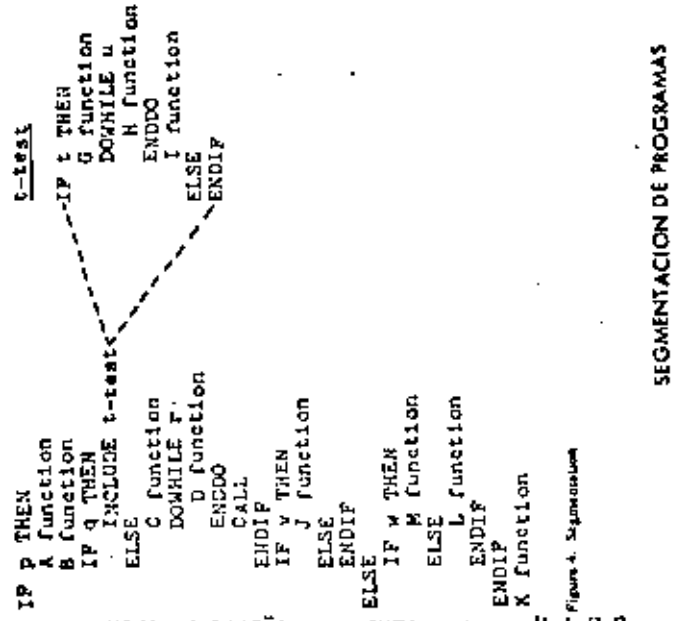


Figura 4. Segmentación

SEGMENTACION DE PROGRAMAS

U.N.A.M. | HOJA DE DISEÑO-DOCUMENTACION | H I P O | C E C A F I

SISTEMA:	AUTOR:	HOJA DE
SUBSISTEMA:	REFERENCIA:	FECHA:
APLICACION:	FUNCION:	

ENTRADA	PROCESO	SALIDA
Cualquier problema de programación Programación estructurada Pseudo código.	Reglas para la descomposición de un programa: - Máximo 1 hoja de pseudo código por segmento. - Contiene estructuras completas. - Si se requiere probar condiciones, el segmento contendrá solamente: - IFTHENELSE y/o - DOWHILE y/o - Llamadas a otros segmentos. - Si no requiere probar condiciones, el segmento contendrá solamente figuras de SEQUENCE.	Un programa propio formado por una jerarquía de segmentos. A su vez cada segmento es un programa propio y su lógica es narrada con pseudo código.

CARACTERISTICAS DE UN PROGRAMA SEGMENTADO

- PROBLEMA MAXIMO: 50 POSTULADOS (6 UNA PANTALLA)
  INDEPENDIENTEMENTE DEL TAMAÑO DEL PROGRAMA

- INCREMENTA EL USO DE SEGMENTOS GENERALES
  MENOS ERRORES

- DESCOMPOSICION EN FUNCIONES
  AUTOMATICA

- FACILITA PRUEBA PARCIAL DE UN PROGRAMA
  USO DE CAROS

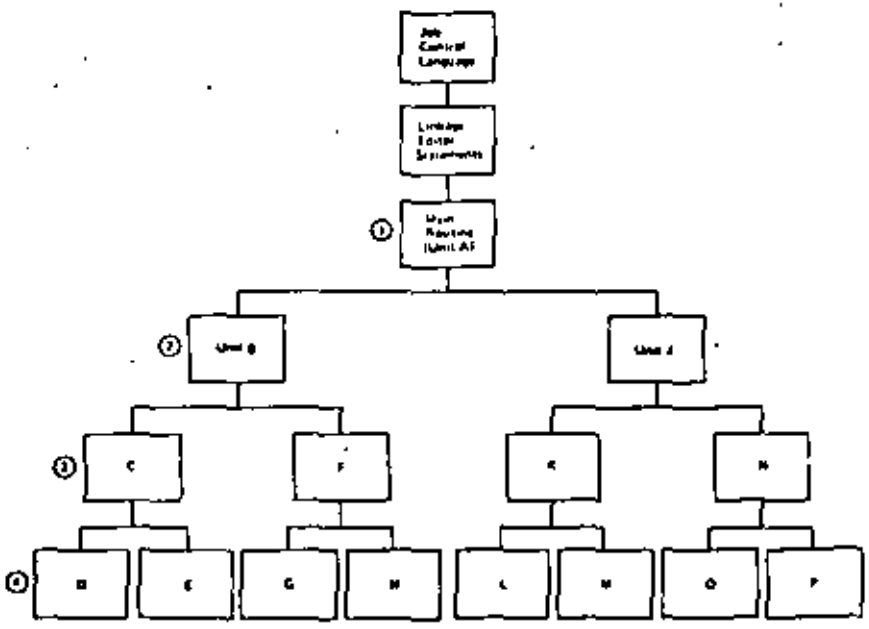
- FACILITA LOCALIZACION DE FUNCIONES

CONVENCIONES DEL DESARROLLO DESCENDENTE

. LA LOGICA MAS EXTERNA SE DESARROLLA EN SU TOTALIDAD ANTES DE INICIAR EL DESARROLLO DE LOGICAS INTERNAS.

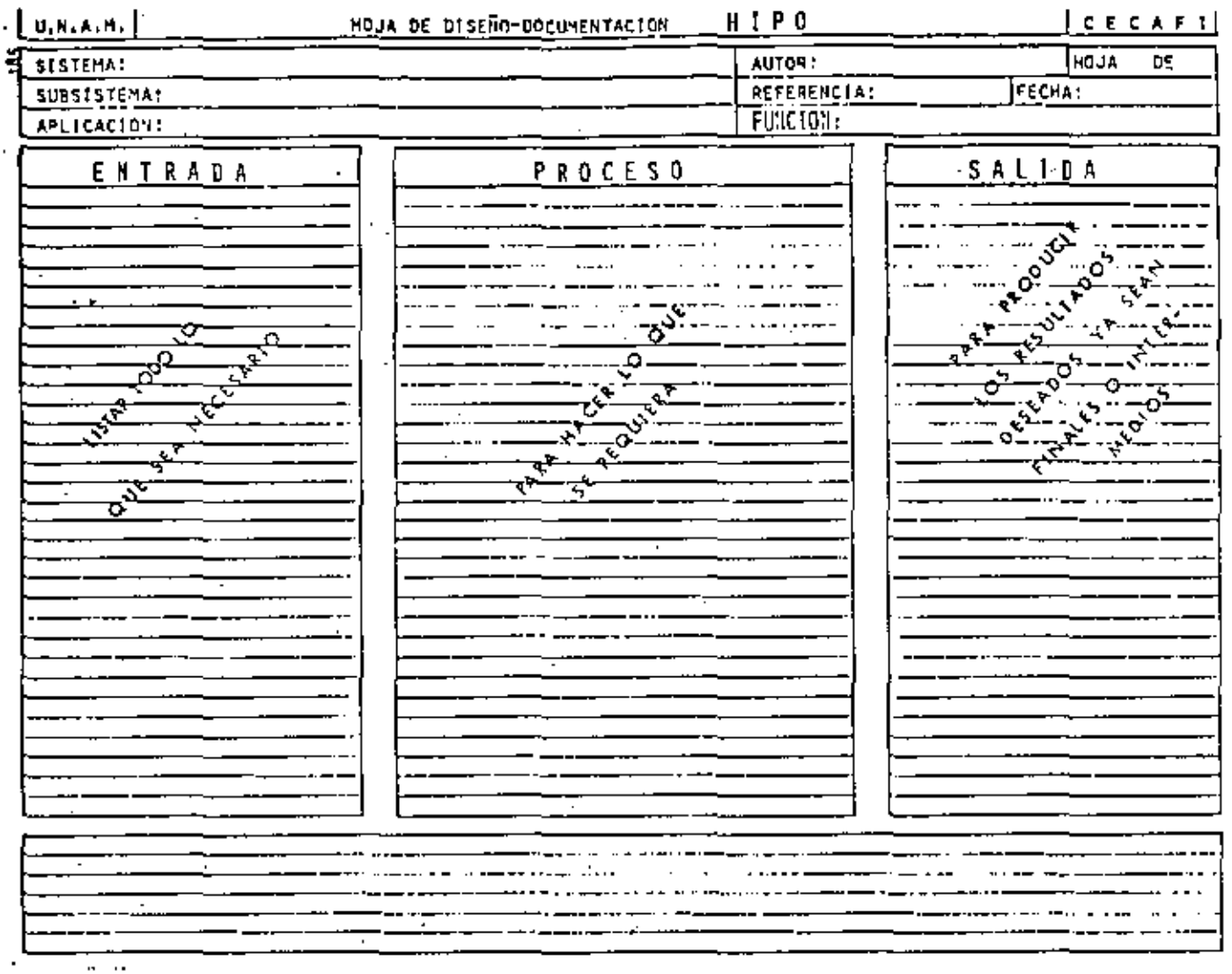
. EL PROCESO ANTERIOR SE REPITE A TODOS LOS NIVELES DE LOGICA DEL PROGRAMA.

. AL PROBAR LOGICAS EXTERNAS, COLOCAR MENSAJES DEL TIPO "LLAMADA CORRECTA A RUTINA UNO" EN LOGICAS INTERNAS POR DESARROLLAR.



VENTAJAS DEL DESARROLLO DESCENDENTE

- APLICABLE A PROGRAMAS Y A SISTEMAS A CUALQUIER NIVEL.
- NO SE REQUIEREN PROGRAMAS MANEJADORES NI CREAR DATOS FICTICIOS PARA PRUEBAS.
- LOS DATOS DE PRUEBA SE VAN CREANDO JUNTO CON EL DESARROLLO DEL PROGRAMA.
- NO EXISTEN PROBLEMAS DE ACOPLAMIENTO EN LLAMADAS A RUTINAS.
- EL TEOREMA DE LA ESTRUCTURA GARANTIZA LA EXISTENCIA DEL MODULO POR DESARROLLAR.
- EL PROGRAMA ADQUIERE LA ESTRUCTURA DE UN ARBOL BIEN DEFINIDO.
- LAS RAMAS DEL ARBOL PUEDEN DESARROLLARSE EN FORMA TOTALMENTE INDEPENDIENTE (FACILITA EL TRABAJO EN EQUIPO).
- ES FACIL LOCALIZAR RUTINAS QUE SE REPITEN.



HIPO (HIERARCHY PLUS INPUT-PROCESS-OUTPUT)

(JERARQUIA MAS ENTRADA-PROCESO-SALIDA)

OBJETIVO:

DOCUMENTAR PROGRAMAS Y SISTEMAS BASANDOSE EN LA JERARQUIA DE UN PROCESO E INDICANDO SUS ENTRADAS Y SUS SALIDAS.

VENTAJAS:

- PUEDE SERVIR COMO GUIA EN EL DESARROLLO DE UN PROGRAMA ó SISTEMA.
- PERMITE TENER UNA VISION GLOBAL ó BIEN CONSULTAR EL MENOR DE LOS DETALLES.
- EL MANTENIMIENTO AFECTA SOLO A PARTES ESPECIFICAS DEL DOCUMENTO.
- EVITA LA INTRODUCCION DE LINEAS "PAJA" DE UN TEXTO CONVENCIONAL.
- HACE DESTACAR LOS PUNTOS IMPORTANTES EN FORMA AUTOMATICA

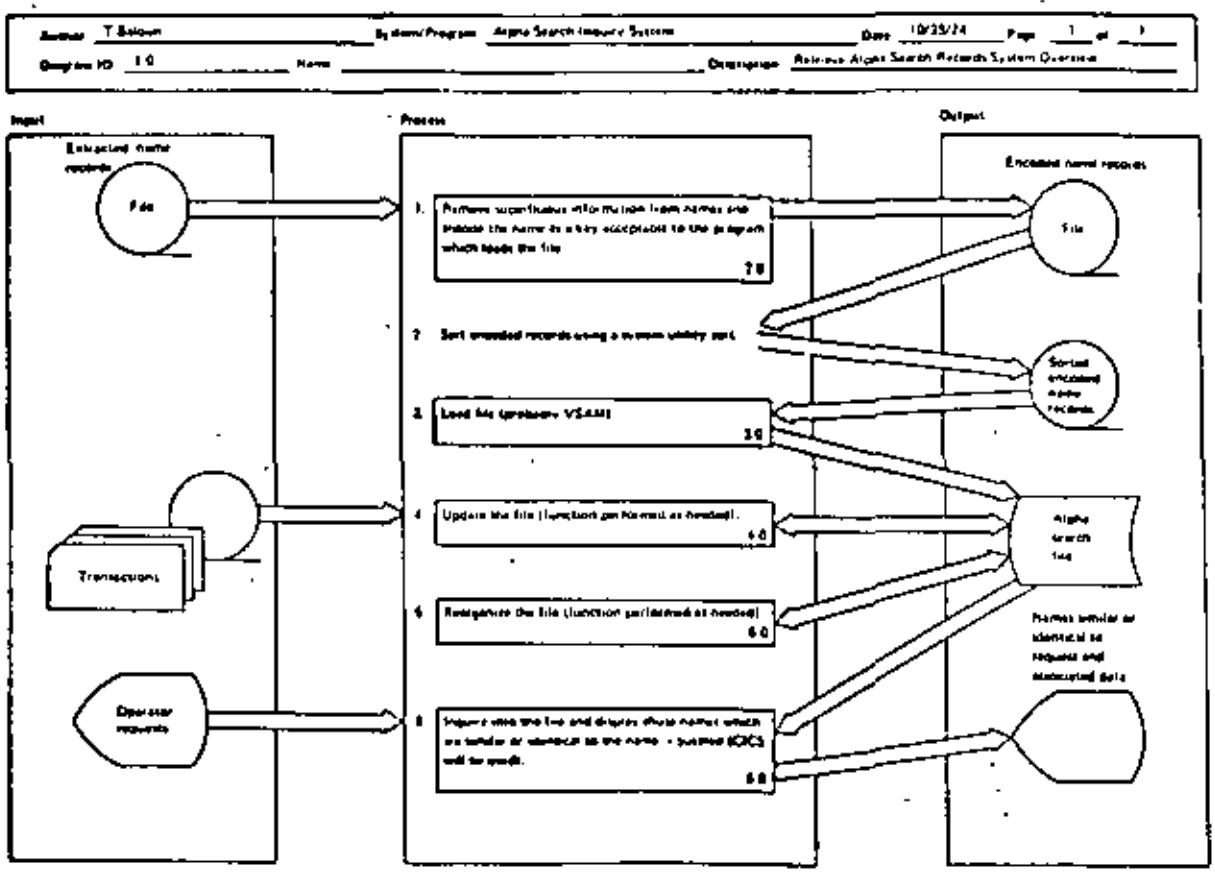


Figure 10 Alpha Search Inquiry System overview diagram (1.0)

BIBLIOTECAS DE SOPORTE

OBJETIVO:

IDENTIFICAR Y CATALOGAR RUTINAS TIPO PARA USO POSTERIOR.

VENTAJAS:

- MINIMIZA LA MULTIPLICACION DE ESFUERZOS.
- FACILITA LA COMPRENSION DE OTROS PROGRAMAS
- AUMENTA LA CAPACIDAD DE DESARROLLO.
- AUMENTA LA CONFIANZA EN EL NUEVO PROGRAMA.

RESUMEN DE RECOMENDACIONES EN PRØGRAMACION ESTRUCTURADA

PARA BASIC TRS 80

- AGRUPAR EL MAXIMO DE INSTRUCCIØNES PØR LINEA FISICA
- COLOCAR SOLO UNA INSTRUCCION POR LINEA LØGICA
- UTILIZAR SOLO ESTRUCTURAS LOGICAS (RECUERDE: EL GØ TØ NUNCA ES NECESARIØ)
- USAR LAS CONVENCIØNES DE CØDIFICACIØN
- USAR SANGRIA EN FØRMA ESTRICTA
- NO CODIFICAR MAS DE TRES NIVELES DE LØGICA SIMULTANEØS
- LIMITAR LØGICAS CØMPLETAS A UNA PANTALLA (26 LINEAS)
- COLOCAR DATA AL FINAL
- INCLUIR CØMENTARIOS REM ó
- UTILIZAR NØMBRES DE VARIABLES LOGICOS DE 2 CARACTERES (CUIDADO PALABRAS RESERVADAS DENTRØ)
- EXCLUIR EL USO DEL GØ TØ
- ACOPLAR ESTAS PRACTICAS CØN PSEUDOCØDIGO, SEGMENTACION, DESARROLLO DESCENDENTE, HIPØ Y LAS RECOMENDACIONES PRØPIAS DEL LEVEL II
- RECOMENDACION FINAL
- ADOPTE ESTAS PRACTICAS PAULATINAMENTE Y RECUERDE "ENTRE MAS RIGIDA ES UNA REGLA, ES MAS INUTIL"

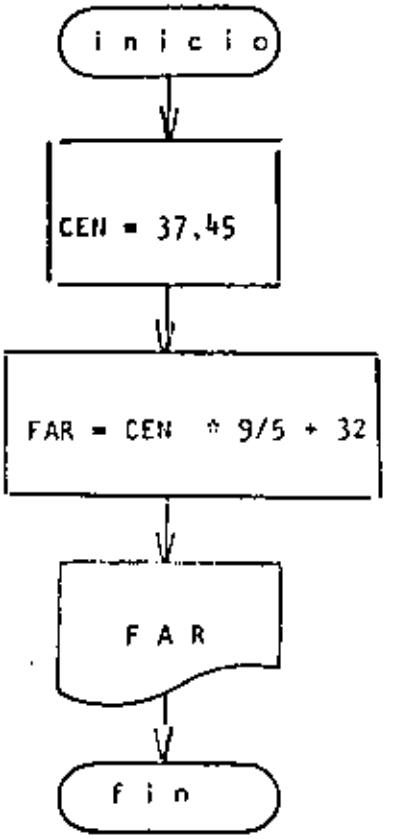
DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA U.N.A.M.

LENGUAJE DE PROGRAMACION BASIC - CON APLICACIONES
(PRIMERA PARTE)

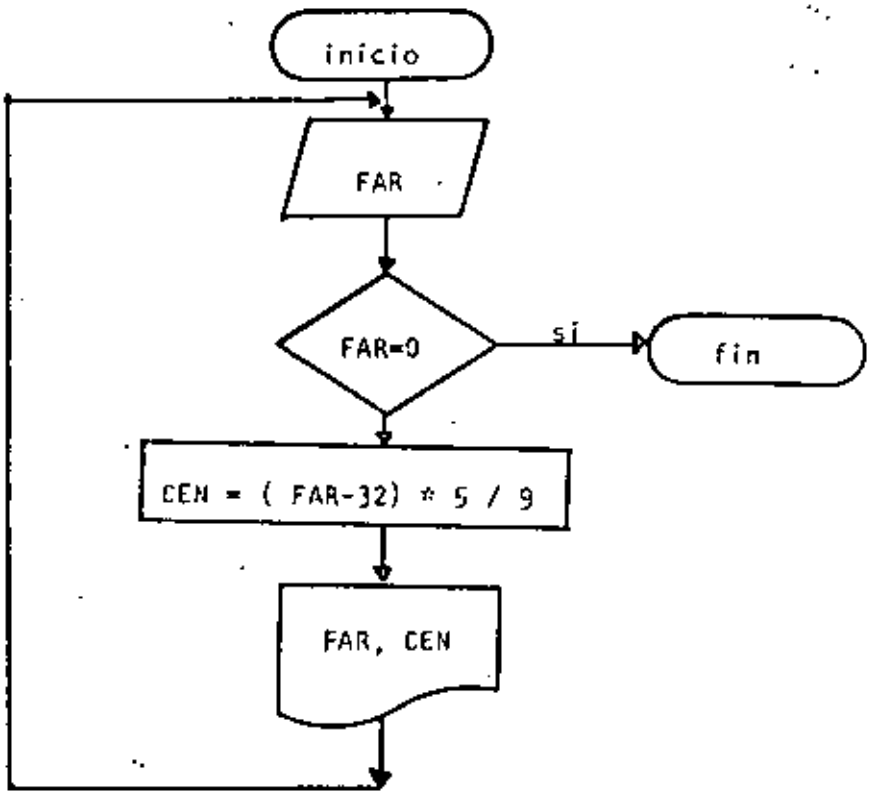
E J E M P L O S

FEBRERO, 1983

Palacio de Minería Calle de Tacuba 5 primer piso Deleg. Cuauhtemoc 06000 México, D.F. Tel.: 521-40-20 Apdo. Postal M-2285



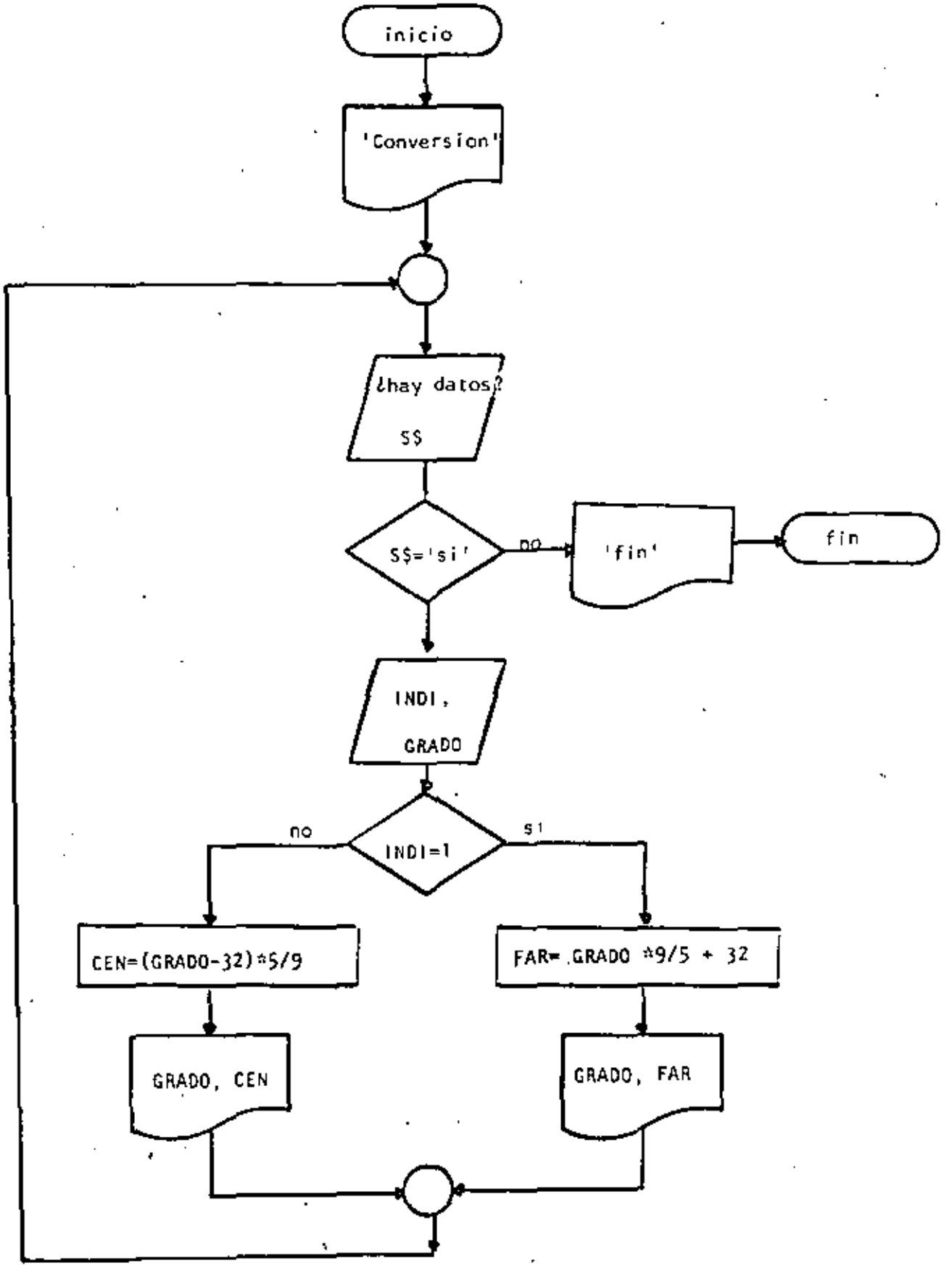

```
1  REM -------UNO-------
10 REM  CONVERSION DE GRADOS CENTIGRADOS A GRADOS FARENHEIT
20 REM
30 CEN=37.45
40 FAR=CEN*9/5+32
50 PRINT FAR
60 END


Ready
>RUN
 99.41
Ready
```




```
'$ RUN:------DOS------
 ) REM CONVERSION DE GRADOS FARENHEIT A GRADOS CENTIGRADOS
20 REM
40 PRINT "CONVERSION DE GRADOS FARENHEIT A GRADOS CENTIGRADOS"
50 PRINT
60 'DOWHILE  FAR SEA <> DE CERO
70 INPUT "FARENHEIT";FAR
80 IF FAR = 0 THEN 130
90       CEN=(FAR-32)*5/9
100      PRINT FAR;" GRADOS FARENHEIT SON ";CEN;" GRADOS CENTIGRADOS"
110 GOTO 70
120 'ENDDO
130 PRINT "FIN DEL PROGRAMA"
140 END
```

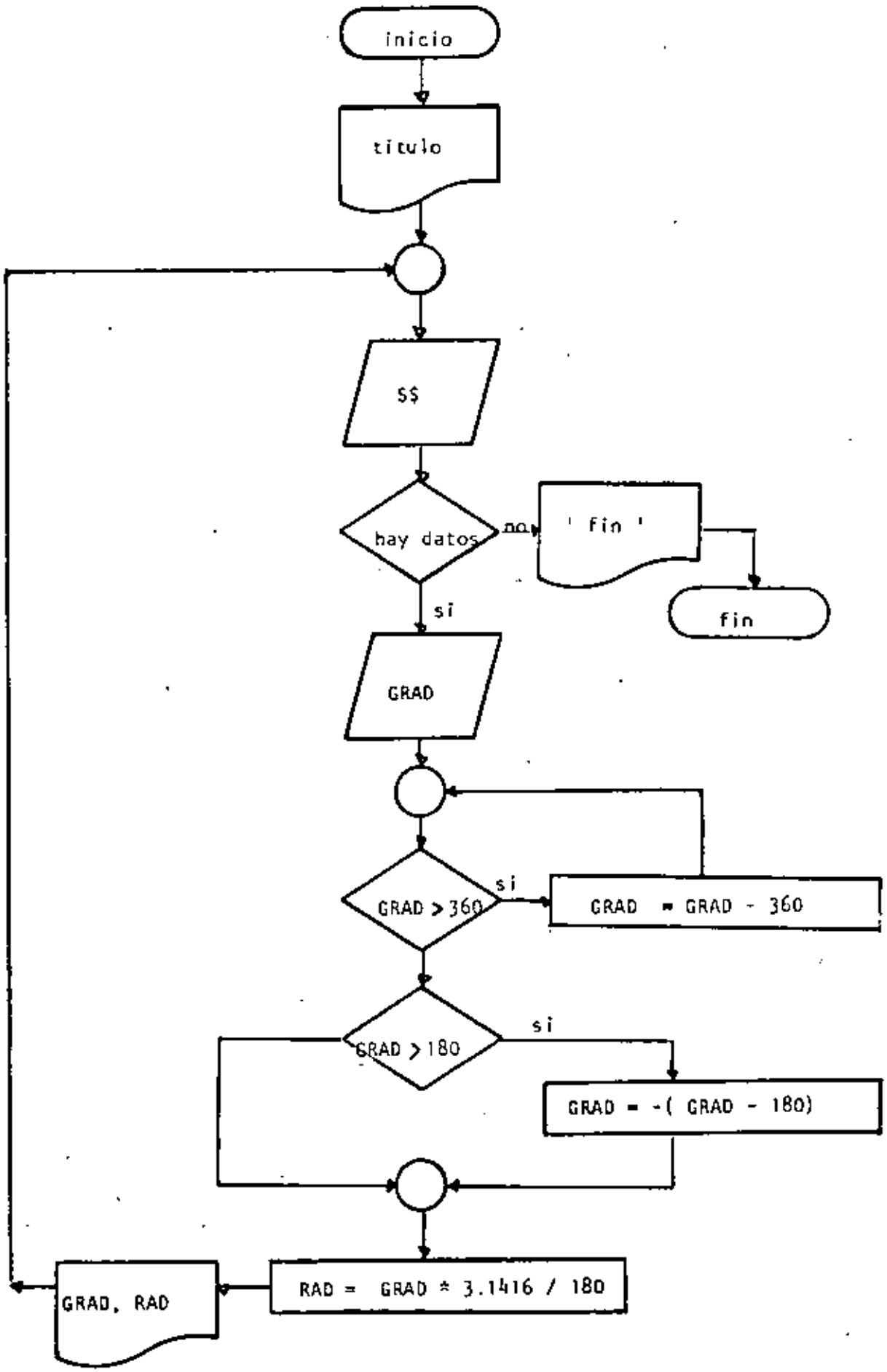
```
Ready
>RUN
CONVERSION DE GRADOS FARENHEIT A GRADOS CENTIGRADOS

FARENHEIT? 34
 34  GRADOS FARENHEIT SON  1.11111  GRADOS CENTIGRADOS
FARENHEIT? 56
 56  GRADOS FARENHEIT SON  13.3333  GRADOS CENTIGRADOS
FARENHEIT? -45
-45  GRADOS FARENHEIT SON -42.7778  GRADOS CENTIGRADOS
FARENHEIT? 0
FIN DEL PROGRAMA
Ready
```

CONVERSION ENTRE GRADOS CENTIGRADOS Y GRADOS FARENHEIT



```
10 REM-------TRES------
20 REM CONVERSION ENTRE GRADOS CENTIGRADOS Y GRADOS FARENHEIT
 0 REM
40 PRINT "CONVERSION DE GRADOS CENTIGRADOS A FARENHEIT O VICEVERSA"
50 PRINT
60 'DOWHILE HAYA DATOS
70 INPUT "HAY DATOS (SI O NO)";S$
80 IF S$ <> "SI" THEN 230
90      INPUT "TIPO (1=CENTIGRADOS, <>1=FARENHEIT) Y GRADOS";INDI,GRAD
100     IF INDI = 1 THEN 120
110                 GOTO 160
120 '      THEN
130             FAR=GRAD*9/5+32
140             PRINT GRAD;"GRADOS CENTIGRADOS SON ";FAR;" GRADOS FARENHEIT"
150          GOTO 200
160 '      ELSE
170             CEN=(GRAD-32)*5/9
180             PRINT GRAD;" GRADOS FARENHEIT SON ";CEN;" GRADOS CENTIGRADOS"
190 '   ENDIF
200 GOTO 60
210 'ENDDO
220 '
230 PRINT "FIN DEL PROCESO"
240 END


Ready

>RUN
CONVERSION DE GRADOS CENTIGRADOS A FARENHEIT O VICEVERSA

HAY DATOS (SI O NO)? SI
TIPO (1=CENTIGRADOS, <>1=FARENHEIT) Y GRADOS? 1,34
 34 GRADOS CENTIGRADOS SON  93.2  GRADOS FARENHEIT
HAY DATOS (SI O NO)? SI
TIPO (1=CENTIGRADOS, <>1=FARENHEIT) Y GRADOS? 2,93.2
 93.2  GRADOS FARENHEIT SON  34  GRADOS CENTIGRADOS
HAY DATOS (SI O NO)? SI
TIPO (1=CENTIGRADOS, <>1=FARENHEIT) Y GRADOS? 0,65
 65  GRADOS FARENHEIT SON  18.3333  GRADOS CENTIGRADOS
HAY DATOS (SI O NO)? NO
FIN DEL PROCESO
Ready
```

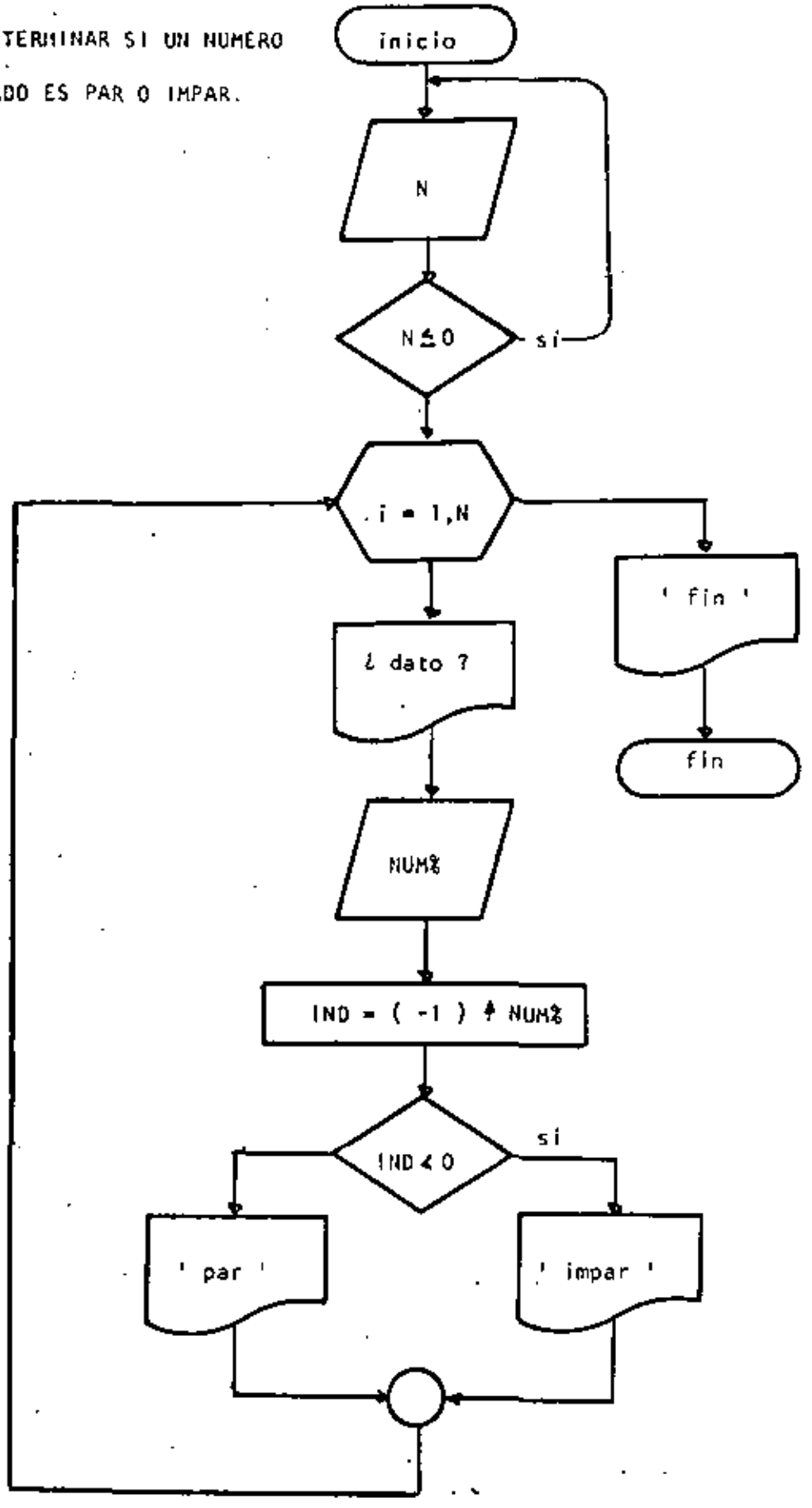
CONVERSION DE GRADOS A RADIANES

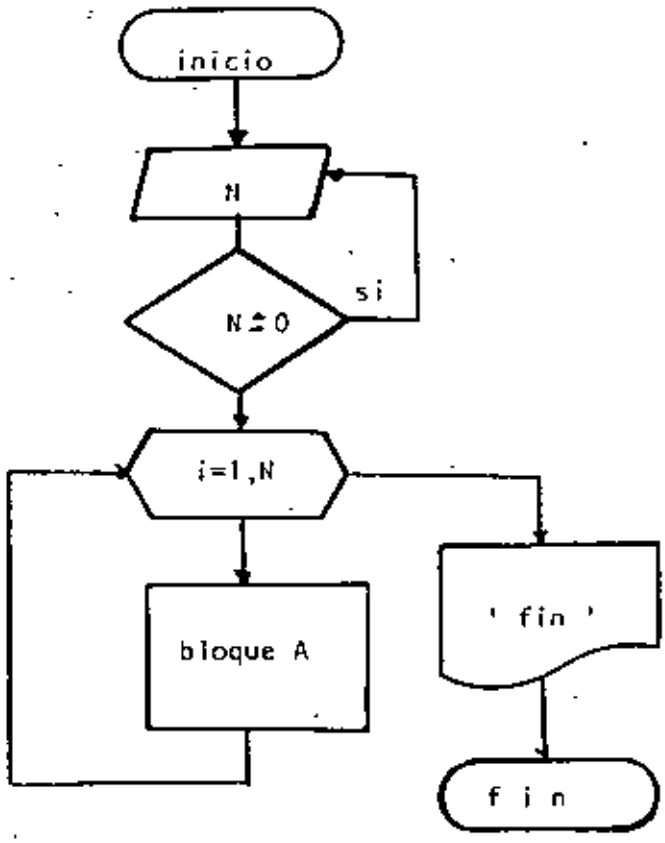


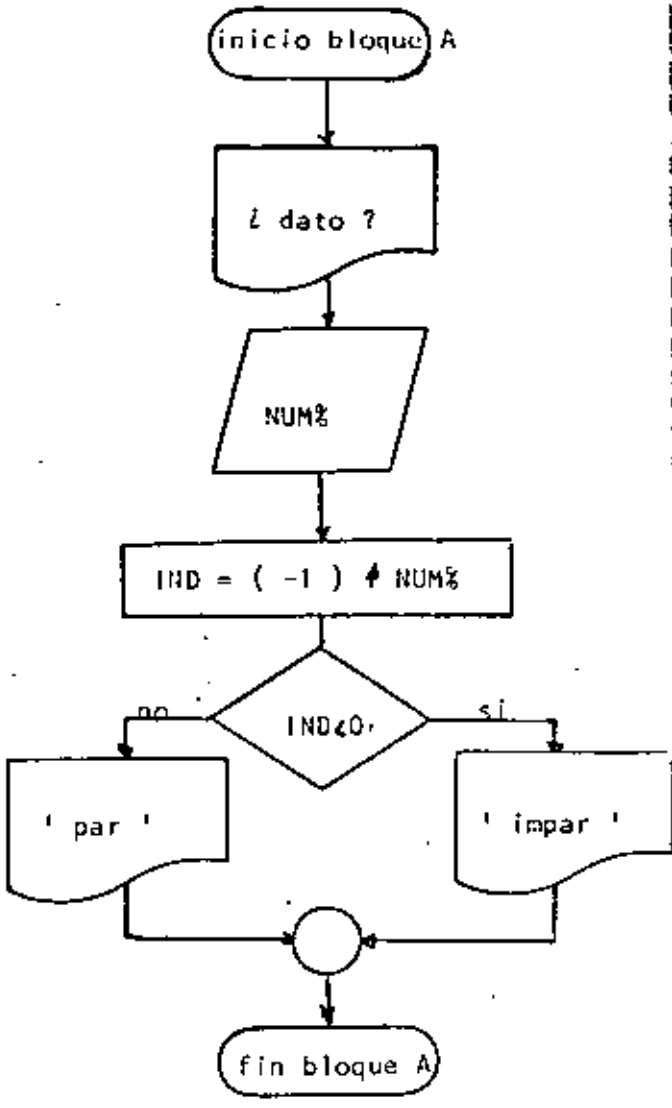
```
10 REM-----CUATRO-------
20 REM CONVERSION DE GRADOS A RADIANES
30 REM
40 PRINT "CONVERSION DE GRADOS A RADIANES":PRINT
50 'DOWHILE HAYA DATOS
60 INPUT "HAY DATOS(SI O NO)";S$
70 IF S$ <> "SI" THEN 220
80       INPUT"GRADOS";GRAD
90 '     DOWHILE GRADOS ES MAYOR QUE 360 AJUSTAR EL VALOR
100      IF ABS(GRAD) < 360 THEN 140
110              GRAD=GRAD - SGN(GRAD)*360
120      GOTO 100
130      'ENDDO
140      ' SE TRABAJA ENTRE -180 Y + 180
150      IF GRAD > 180 THEN GRAD =-(GRAD-180)
160      RAD=GRAD*3.1416/180
170      PRINT GRAD;" GRADOS SON ";RAD;" RADIANES"
180  GOTO 60
190 'ENDDO
200 '
210 'FIN
220 PRINT "FIN DEL PROCESO"
230 END
```

```
>RUN
CONVERSION DE GRADOS A RADIANES

HAY DATOS(SI O NO)? SI
GRADOS? 34
 34  GRADOS SON  .593413  RADIANES
HAY DATOS(SI O NO)? SI
GRADOS? 194
-14  GRADOS SON -.244347  RADIANES
HAY DATOS(SI O NO)? SI
GRADOS? 567
-27  GRADOS SON -.47124  RADIANES
HAY DATOS(SI O NO)? SI
GRADOS? 120
 120  GRADOS SON  2.0944  RADIANES
HAY DATOS(SI O NO)? NO
FIN DEL PROCESO
Ready
```

DETERMINAR SI UN NUMERO DADO ES PAR O IMPAR.



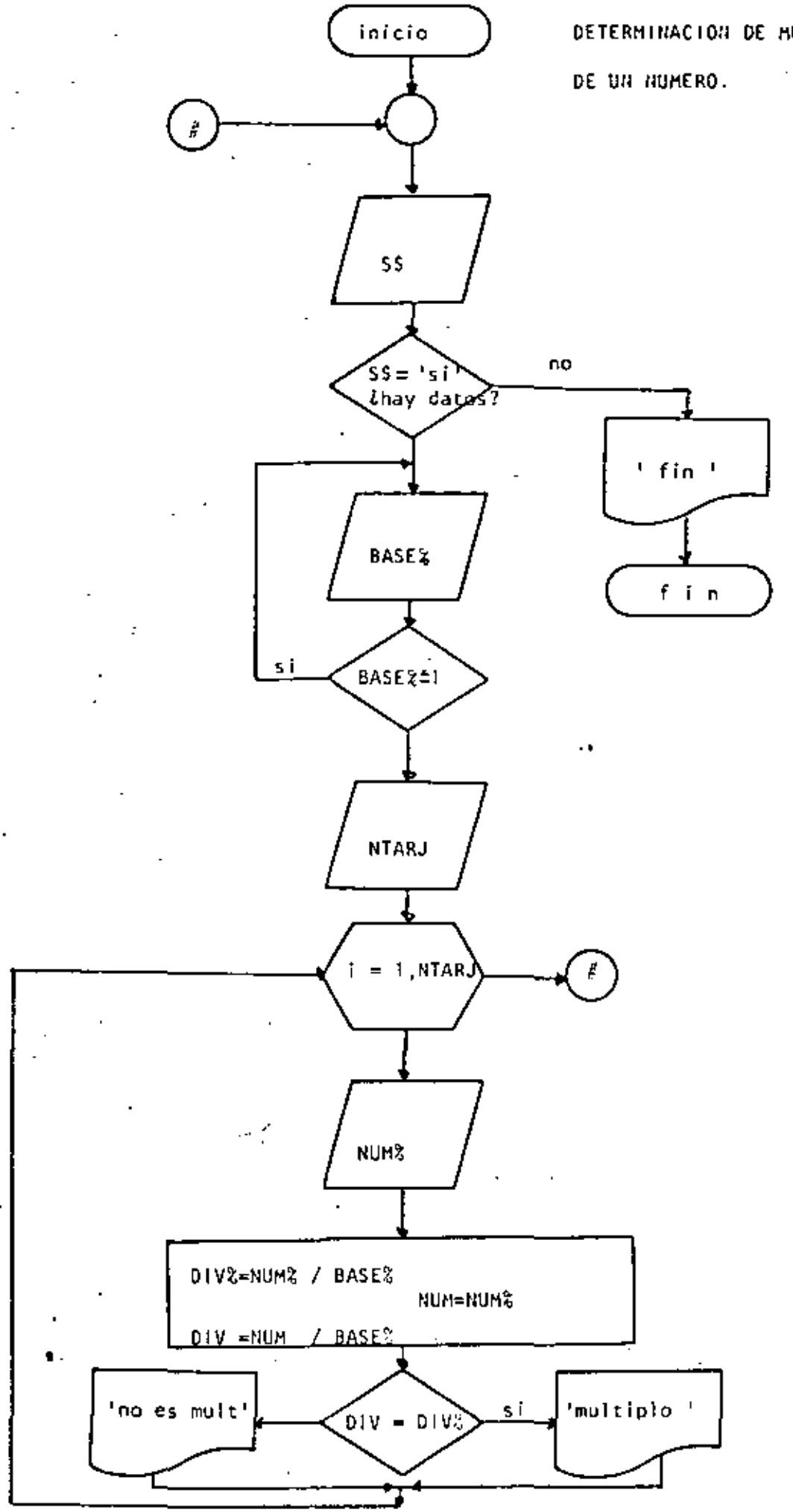





```
1   REM------CINCO-------
10 REM DETERMINACION DE NUMEROS PARES E IMPARES
20 REM
30 INPUT "TOTAL DE DATOS A PROPORCIONAR";N
40 IF N<=0 THEN 30
50 FOR I=1 TO N
60       PRINT "DAME EL DATO NUMERO:";I;" ";
70       INPUT NUM%
80       IND%=(-1)↑NUM%
90       IF IND% <0 THEN PRINT "EL NUMERO ";NUM%;" ES IMPAR "
                    ELSE PRINT "EL NUMERO ";NUM%;" ES
 PAR "
100 NEXT I
110 PRINT "FIN DEL PROGRAMA"
120 END
```

```
>RUN
TOTAL DE DATOS A PROPORCIONAR? 3
DAME EL DATO NUMERO: 1  ? 3
EL NUMERO  3  ES IMPAR
DAME EL DATO NUMERO: 2  ? 6
EL NUMERO  6  ES PAR
DAME EL DATO NUMERO: 3  ? 87
EL NUMERO  87  ES IMPAR
FIN DEL PROGRAMA
Ready
```

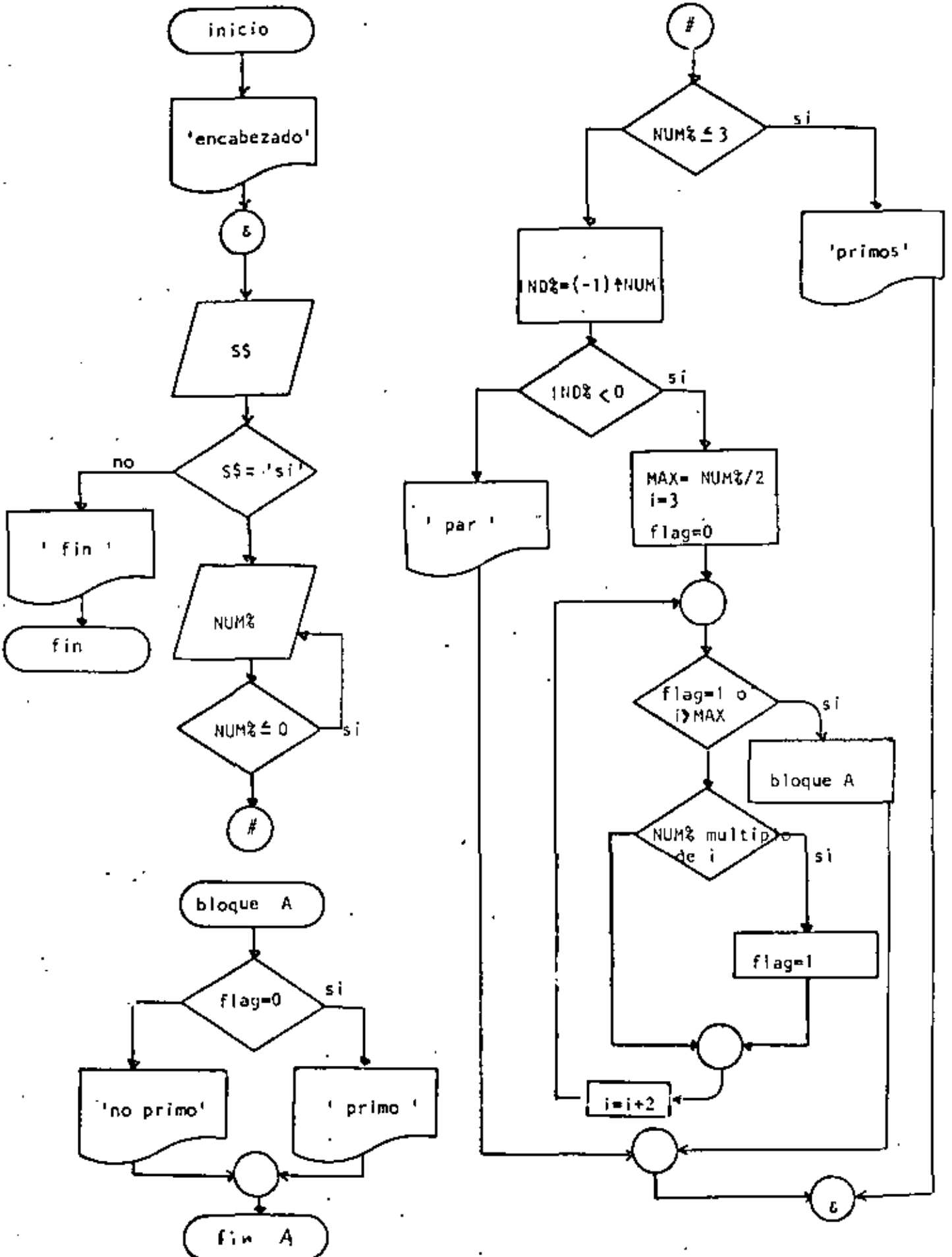
DETERMINACION DE MULTIPLOS DE UN NUMERO.



```
1  REM-----SEIS------
10 REM DETERMINACION DE MULTIPLOS DE UN NUMERO
20 REM
30 '
40 'DOWHILE HAYA DATOS
50  INPUT"HAY DATOS (SI O NO)";S$
60  IF S$ <> "SI" THEN 170
70      INPUT "BASE";BASE%: IF BASE% <=1 THEN 70
80      INPUT "CANTIDAD DE DATOS";NTARJ
90      FOR I=1 TO NTARJ
100             PRINT "DATO NUMERO ";I;:INPUT NUM%
110             DIV%=NUM%/BASE%
120             NUM=NUM%
130             DIV=NUM/BASE%
140             IF DIV=DIV% THEN PRINT NUM%;" SI ES MULTIPLO DE ";BASE%
                        ELSE PRINT NUM%;" NO ES
MULTIPLO DE ";BASE%
150     NEXT I
160 GOTO 50
170 PRINT "FIN DEL PROCESO"
180 END
>RUN
HAY DATOS (SI O NO)? SI
BASE? 4
CANTIDAD DE DATOS? 6
DATO NUMERO  1 ? 3
 3  NO ES MULTIPLO DE  4
DATO NUMERO  2 ? 5
 5  NO ES MULTIPLO DE  4
DATO NUMERO  3 ?
 5  NO ES MULTIPLO DE  4
DATO NUMERO  4 ?
 5  NO ES MULTIPLO DE  4
DATO NUMERO  5 ? 4
 4  SI ES MULTIPLO DE  4
DATO NUMERO  6 ? 2
 2  NO ES MULTIPLO DE  4
HAY DATOS (SI O NO)? NO
FIN DEL PROCESO
Ready
```

```
RUN
HAY DATOS (SI O NO)? SI
BASE? 3
CANTIDAD DE DATOS? 5
DATO NUMERO  1 ? 35
 35  NO ES MULTIPLO DE  3
DATO NUMERO  2 ?
 35  NO ES MULTIPLO DE  3
DATO NUMERO  3 ?
 35  NO ES MULTIPLO DE  3
DATO NUMERO  4 ? 9
 9  SI ES MULTIPLO DE  3
DATO NUMERO  5 ? 0
 0  SI ES MULTIPLO DE  3
HAY DATOS (SI O NO)? NO
FIN DEL PROCESO
Ready
```

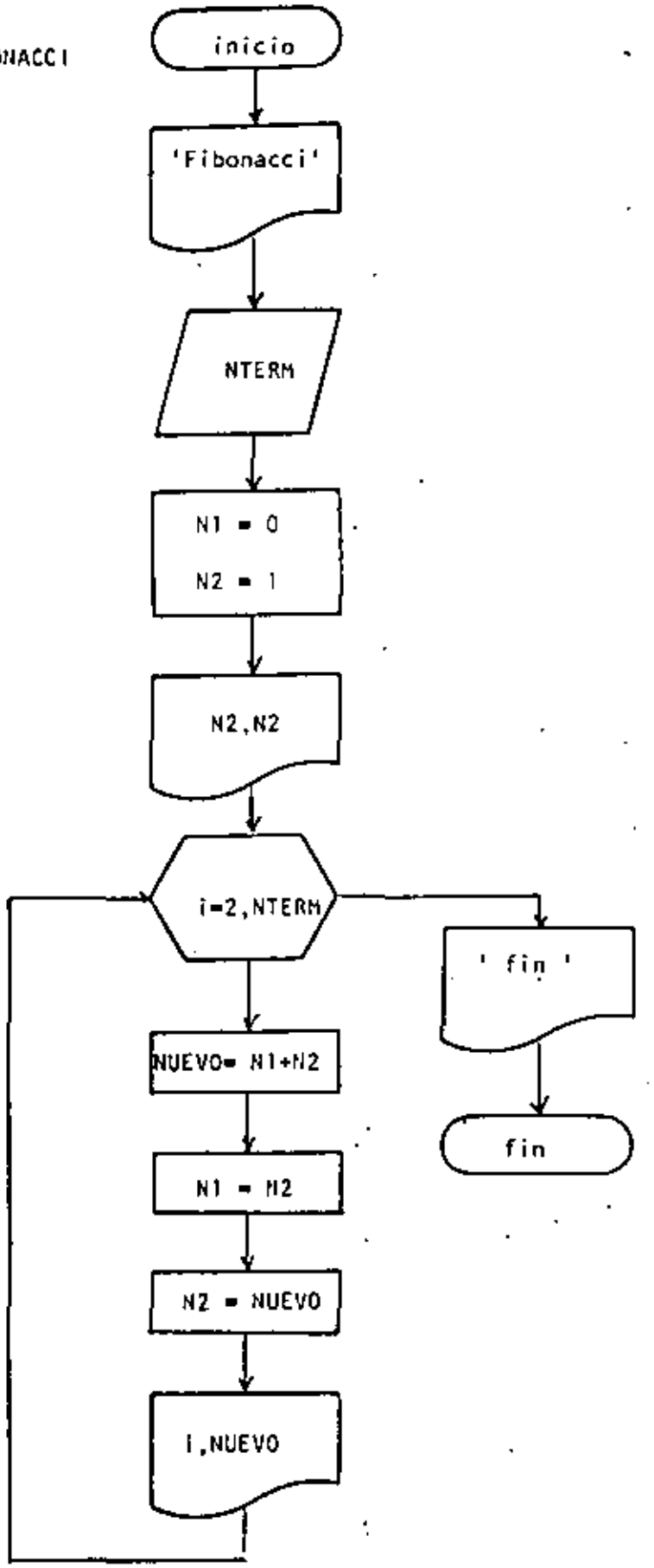
NUMEROS PRIMOS



```
1  REM-----SIETE-----
10 REM NUMEROS PRIMOS
20 REM
30 PRINT TAB(40);'PRIMOS',TAB(50);'NO PRIMOS'
40 'DOWHILE HAYA DATOS
50  INPUT 'HAY DATOS(SI O NO)';S$
60  IF S$ <> 'SI' THEN 400
70       INPUT 'DAME EL DATO ENTERO POSITIVO';NUMX
80       IF NUMX <= 0 THEN 70
90       IF NUMX <= 3 THEN 110
100                   GOTO 150
110      'THEN
120              PRINT TAB(42);NUMX
130              ' 1 2 Y 3 SON PRIMOS
140      GOTO 360
150      'ELSE
160              INDX=(-1)^NUMX
170              IF INDX < 0 THEN 200
180                       GOTO 330
190              'THEN
200                      'NUMERO IMPAR
210                      MAX=NUMX/2
220                      I=3
230                      FLAG=0
240                      IF FLAG=1 OR I>MAX THEN 300
250                      'DOWHILE SE DETERMINE SI ES DIVISIBLE
260                              IF NUMX=FIX(NUMX/I)*I THEN FLAG=1
270                              I=I+2
280                      GOTO 240
290                      'ENDWHILE
300                      IF FLAG=0 THEN PRINTTAB(42);NUMX
                                   ELSE PRINTTAB(52);
NUMX
310                      GOTO 350
320              'ELSE
330                      PRINT TAB(52);NUMX;'ES NUMERO PAR
340              'ENDIF
350      'ENDIF
360 'ENDWHILE
370 GOTO 40
380 '
390 'FIN DEL PROCESO
400 PRINT 'FIN DE LOS NUMEROS PRIMOS'
410 END
```

```
>RUN
                                                PRIMOS     NO PRIMOS
HAY DATOS(SI O NO)? SI
DAME EL DATO ENTERO POSITIVO? 45
                                                           45
HAY DATOS(SI O NO)? SI
DAME EL DATO ENTERO POSITIVO? 7
                                                  7
HAY DATOS(SI O NO)? SI
DAME EL DATO ENTERO POSITIVO? 567
                                                           567
HAY DATOS(SI O NO)? NO
FIN DE LOS NUMEROS PRIMOS
```

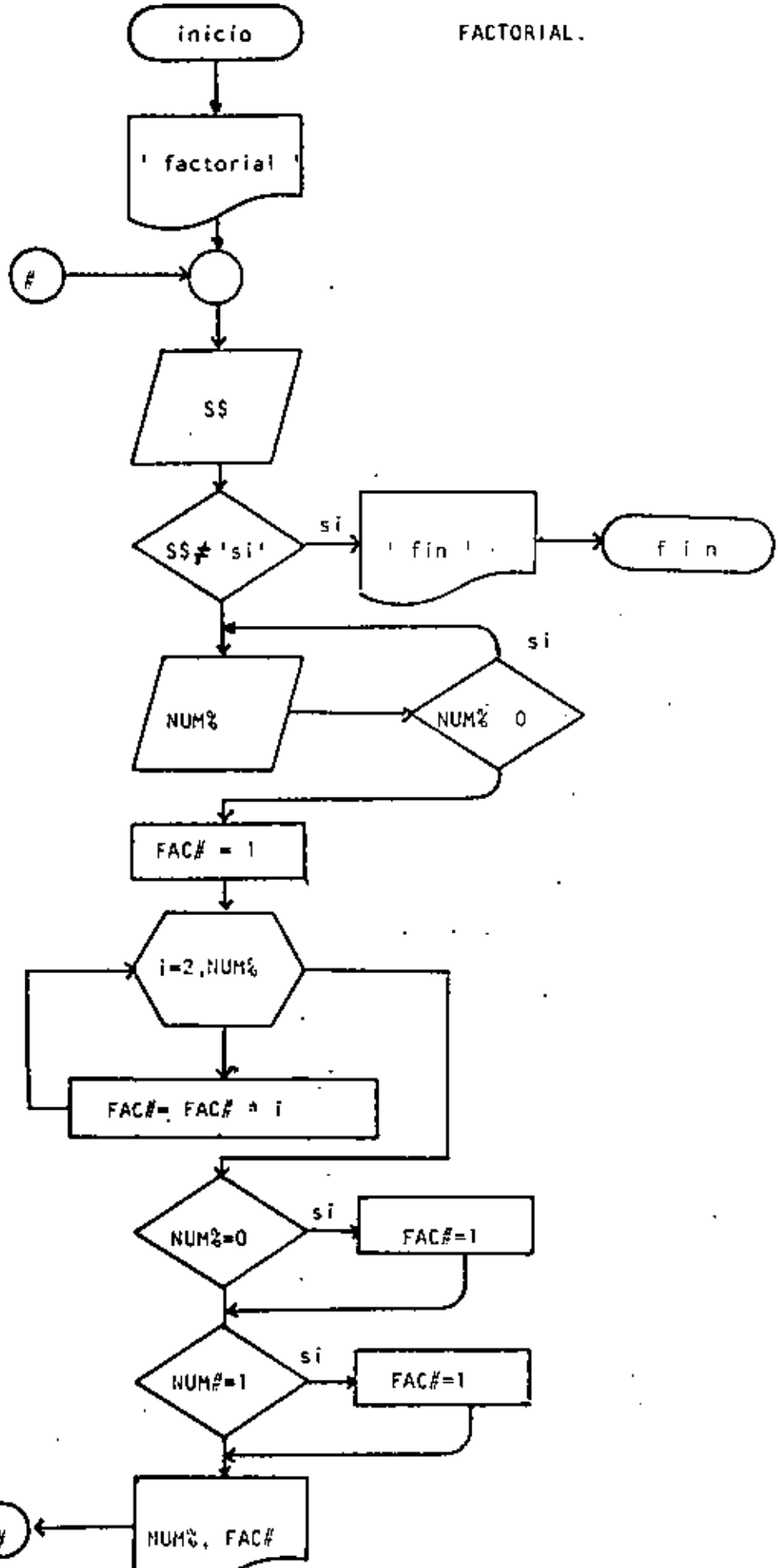
SERIE DE FIBONACCI



```
1  REM----OCHO----
10 REM SERIE DE FIBONACCI
20 REM
30 PRINT "SERIE DE FIBONACCI"
40 PRINT
50 INPUT"DAME EL NUMERO DE TERMINOS";NTERM
60 N1=0
70 N2=1
80 PRINT"   TERMINO        VALOR "
90 PRINT
100 PRINT USING"     ###     ##########";N2,N2
110 FOR I=2 TO NTERM
120      NUEVO=N1+N2
130      N1    = N2
140      N2    = NUEVO
150      PRINT USING"     ###     ##########";I,NUEVO
160 NEXT I
170 PRINT "FIN DE LA SERIE"
180 END
```

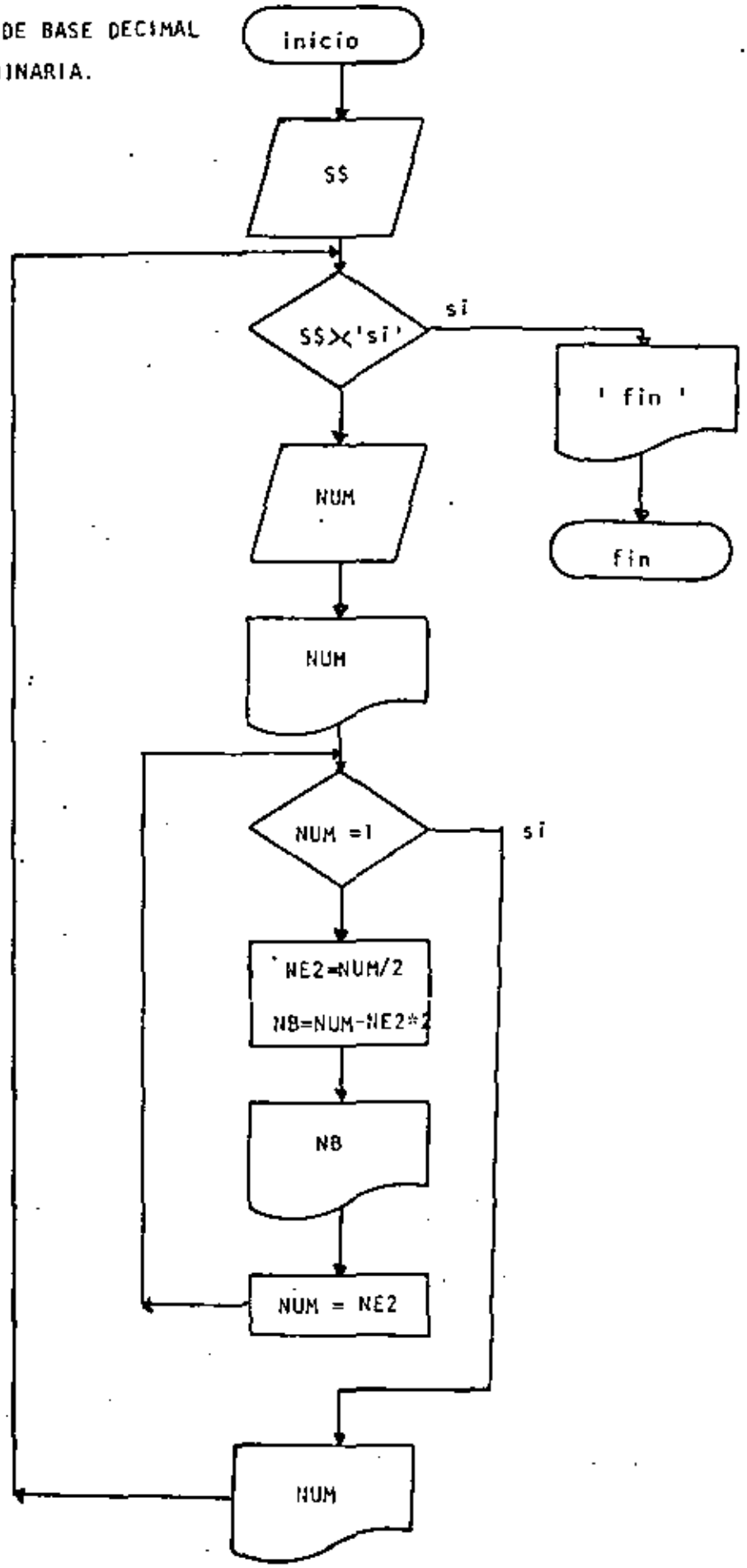
```
Ready

>RUN
SERIE DE FIBONACCI


DAME EL NUMERO DE TERMINOS? 15
   TERMINO        VALOR

       1              1
       2              1
       3              2
       4              3
       5              5
       6              8
       7             13
       8             21
       9             34
      10             55
      11             89
      12            144
      13            233
      14            377
      15            610
FIN DE LA SERIE
Ready
```

FACTORIAL.

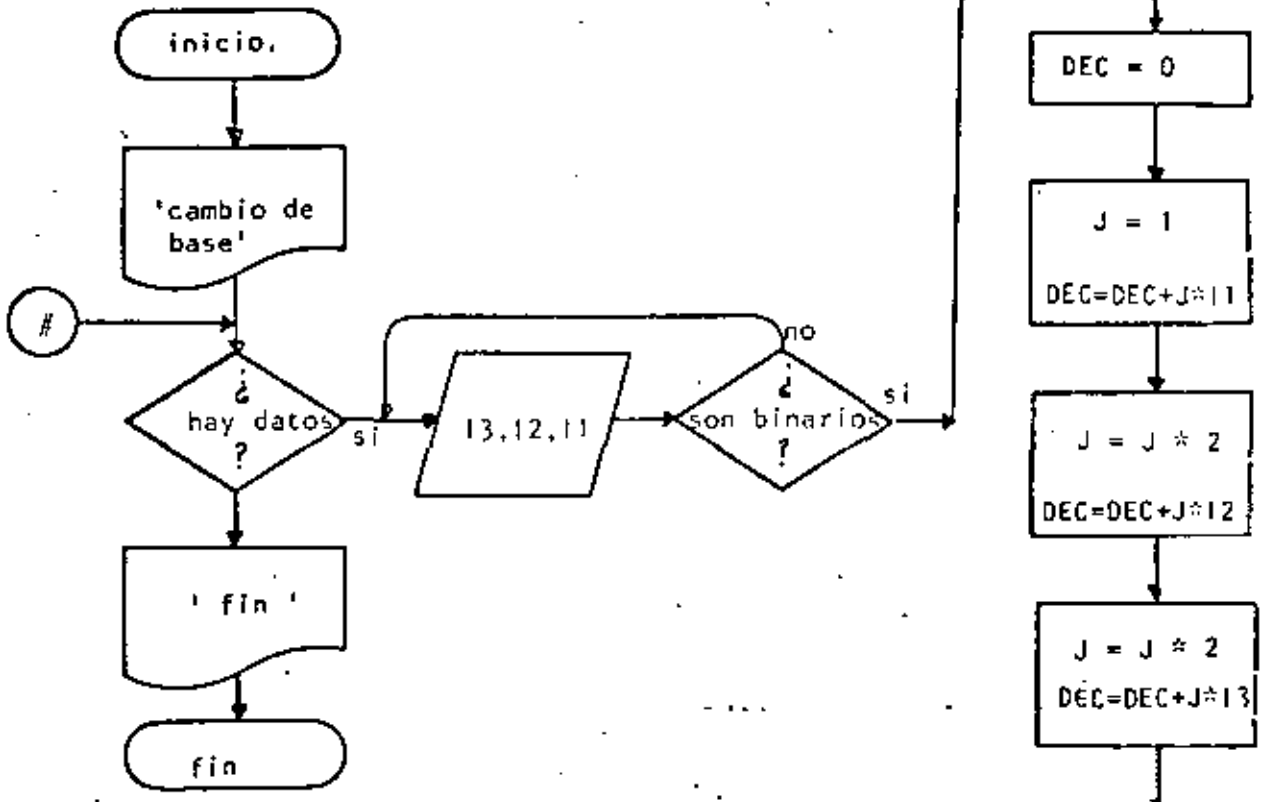

inicio
' factorial '
#
S$
S$≠'si'
si
' fin '
f i n
si
NUM%
NUM% 0
FAC# = 1
i=2,NUM%
FAC#= FAC# * i
NUM%=0
si
FAC#=1
NUM#=1
si
FAC#=1
#
NUM%, FAC#

```
1  REM----NUEVE----
10 REM FACTORIAL
20 REM
30 PRINT "FACTORIAL":PRINT
40 'DOWHILE HAYA DATOS
50  INPUT "HAY DATOS (SI O NO)";S$
60  IF S$ <> "SI" THEN 180
70       INPUT"DAME EL DATO";NUM%
80       IF NUM% < 0 THEN 70
90       FAC#=1
100      FOR I=2 TO NUM%
110              FAC#=FAC#*I
120      NEXT I
130      IF NUM%=0 THEN FAC#=1
140      IF NUM%=1 THEN FAC#=1
150      PRINT "EL FACTORIAL DE ";NUM%;" VALE ";FAC#
160  GOTO 40
170 'ENDDO
180 PRINT "FIN DE FACTORIAL"
190 END
>RUN
FACTORIAL

HAY DATOS (SI O NO)? SI
DAME EL DATO? 3
EL FACTORIAL DE  3  VALE  6
HAY DATOS (SI O NO)? SI
DAME EL DATO? 6
EL FACTORIAL DE  6  VALE  720
HAY DATOS (SI O NO)? SI
DAME EL DATO? 15
EL FACTORIAL DE  15  VALE  1307674368000
HAY DATOS (SI O NO)? SI
DAME EL DATO? 30
EL FACTORIAL DE  30  VALE  2.6525285981219110+32
HAY DATOS (SI O NO)? SI
DAME EL DATO? 32
EL FACTORIAL DE  32  VALE  2.6313083693369360+35
HAY DATOS (SI O NO)? SI
DAME EL DATO? 35
?OV Error in 110
Ready
>RUN
FACTORIAL

HAY DATOS (SI O NO)? SI
DAME EL DATO? 8
EL FACTORIAL DE  8  VALE  40320
HAY DATOS (SI O NO)? NO
FIN DE FACTORIAL
Ready
```

CAMBIO DE BASE DECIMAL
A BINARIA.

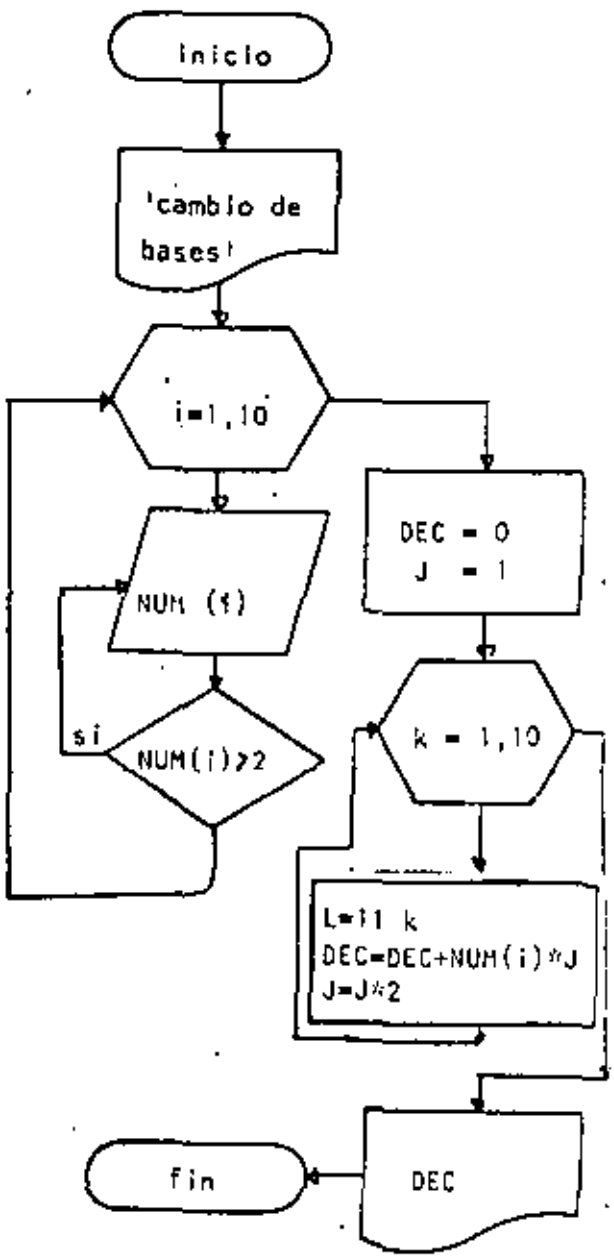


```
10  REM----DIEZ----
20 REM CAMBIO DE BASE DECIMAL A BASE BINARIA
30 REM
40  DEFINT N
50 'DOWHILE HAYA DATOS
60  INPUT 'HAY DATOS (SI.O NO)';S$
70  IF S$ <> 'SI' THEN 220
80       INPUT NUM
90       PRINT 'EL NUMERO ';NUM;' EN BASE DECIMAL ES IGUAL  A '
100      'DOWHILE EL RESIDUO SEA MAYOR QUE UNO
110      IF NUM <= 1 THEN 180
120              NE2=NUM/2
130              NB =NUM - NE2*2
140              PRINT TAB(10);NB
150              NUM=NE2
160      GOTO 110
170 '    ENDDO
180      PRINT TAB(10);NUM;' EN BASE BINARIA'
190 PRINT
200 GOTO 60
210 'ENDDO
220 PRINT 'FIN DEL CAMBIO DE BASES'
230 END
>RUN
HAY DATOS (SI O NO)? SI
? 34
EL NUMERO  34  EN BASE DECIMAL ES IGUAL  A
           0
           1
           0
           0
           0
           1  EN BASE BINARIA

HAY DATOS (SI O NO)? SI
? 9
EL NUMERO  9  EN BASE DECIMAL ES IGUAL  A
           1
           0
           0
           1  EN BASE BINARIA

HAY DATOS (SI O NO)? SI
? 3
EL NUMERO  3  EN BASE DECIMAL ES IGUAL  A
           1
           1  EN BASE BINARIA

HAY DATOS (SI O NO)? SI
? 4
EL NUMERO  4  EN BASE DECIMAL ES IGUAL  A
           0
           0
           1  EN BASE BINARIA

HAY DATOS (SI O NO)? NO
FIN DEL CAMBIO DE BASES
Ready
```

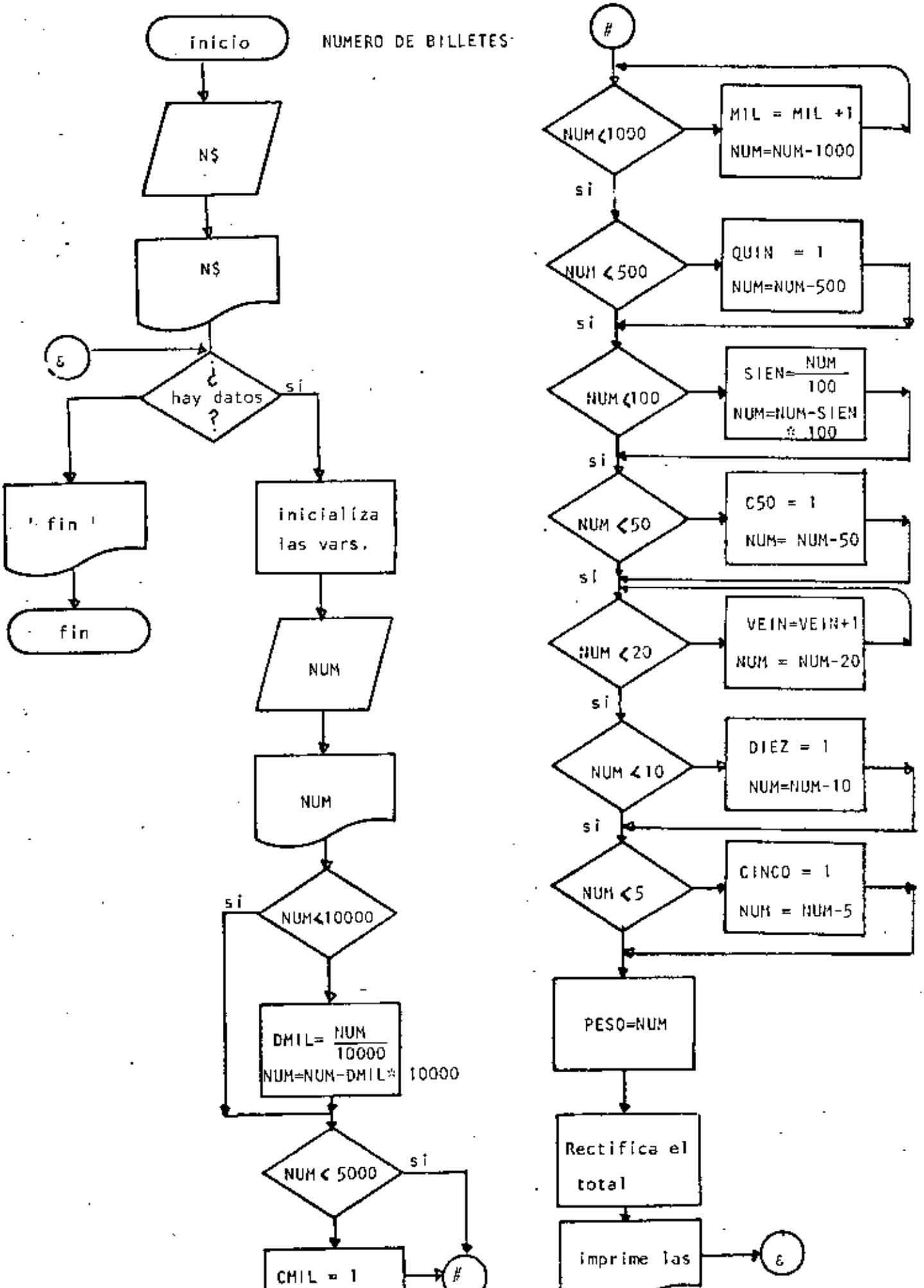



```
10  REM----ONCE----
20 REM CAMBIO DE BASE BINARIA A BASE DECIMAL
30 REM
40  PRINT" CAMBIO DE BASE BINARIA A BASE DECIMAL"
50 'DOWHILE HAYA DATOS
6    INPUT "HAY DATOS (SI O NO)"; S$: IF S$ <> "SI" THEN 190
70       INPUT " LOS TRES DIGITOS BINARIOS ";I3,I2,I1
80     ' IF ABS(I1)>1 OR ABS(I2)>1 OR ABS(I3)>1 THEN 70.
90       DEC=0
100      J=1
110      DEC=DEC+J*I1
120      J=J*2
130      DEC=DEC+J*I2
140      J=J*2
150      DEC=DEC+J*I3
160      PRINT "EL NUMERO BINARIO ";I3;I2;I1;" CORRESPONDE AL ";DEC;" EN BASE DEC
IMAL"
170      GOTO 60
180 'ENDDO
190 PRINT "FIN DE CAMBIO DE BASES"
200 END
>RUN'
 CAMBIO DE BASE BINARIA A BASE DECIMAL
HAY DATOS (SI O NO)? SI
 LOS TRES DIGITOS BINARIOS ? 101
?? 0
?? 0
 LOS TRES DIGITOS BINARIOS ? 1
?? 0
?? 1
!   'UMERO BINARIO   1   0   1  CORRESPONDE AL   5  EN BASE DECIMAL
H.   DATOS (SI O NO)? SI
 LOS TRES DIGITOS BINARIOS ? 1,1,0
EL NUMERO BINARIO   1   1   0  CORRESPONDE AL   6  EN BASE DECIMAL
HAY DATOS (SI O NO)? NO
FIN DE CAMBIO DE BASES
Ready
```

```
10  REM----DOCE----
20 REM CAMBIO DE BASE BINARIA A BASE DECIMAL UTILIZANDO ARREGLOS
30   'M EN BASIC LA DIMENSION POR OMISION ES DE ONCE ( 0 A 10)
40 REM
50 PRINT"CAMBIO DE BASES DE DOS A DIEZ"
60 PRINT
70 'DOWHILE HAYA DATOS
80  INPUT"HAY DATOS(SI O NO)";S$:IF S$ <>"SI" THEN 300
90       PRINT "DAME LOS DIEZ DIGITOS BINARIOS"
100      FOR I=1 TO 10
110              INPUT NUM(I)
120              'DOWHILE EL DIGITO SEA INCORRECTO
130              IF NUM(I) < 2 THEN 170
140                      PRINT "DIGITO ERRONEO, VOLVER A TECLEAR"
150                      GOTO 110
160              'ENDDO
170      NEXT I
180      DEC=0
190      J=1
200      FOR K=1 TO 10
210              L=11-K
220              DEC=DEC+NUM(L)*J
230              J=J*2
240      NEXT K
250      PRINT "EL NUMERO BINARIO ";
260      FOR I=1 TO 10 : PRINT NUM(I);:NEXT I
270      PRINT " ES IGUAL A ";DEC ;" EN DECIMAL"
2        GOTO 70
2.   'ENDDO
300 PRINT" FIN DE CAMBIO DE BASE"
310 END
```




```
>RUN
CAMBIO DE BASES DE DOS A DIEZ

HAY DATOS(SI O NO)? SI
DAME LOS DIEZ DIGITOS BINARIOS
? 0
? 0
? 0
? 0
? 1
? 0
? 1
? 1
?

EL NUMERO BINARIO  0  0  0  0  1  0  1  1  1  0  ES IGUAL A  46  EN DECIMAL
HAY DATOS(SI O NO)? NO
 FIN DE CAMBIO DE BASE
Ready
```

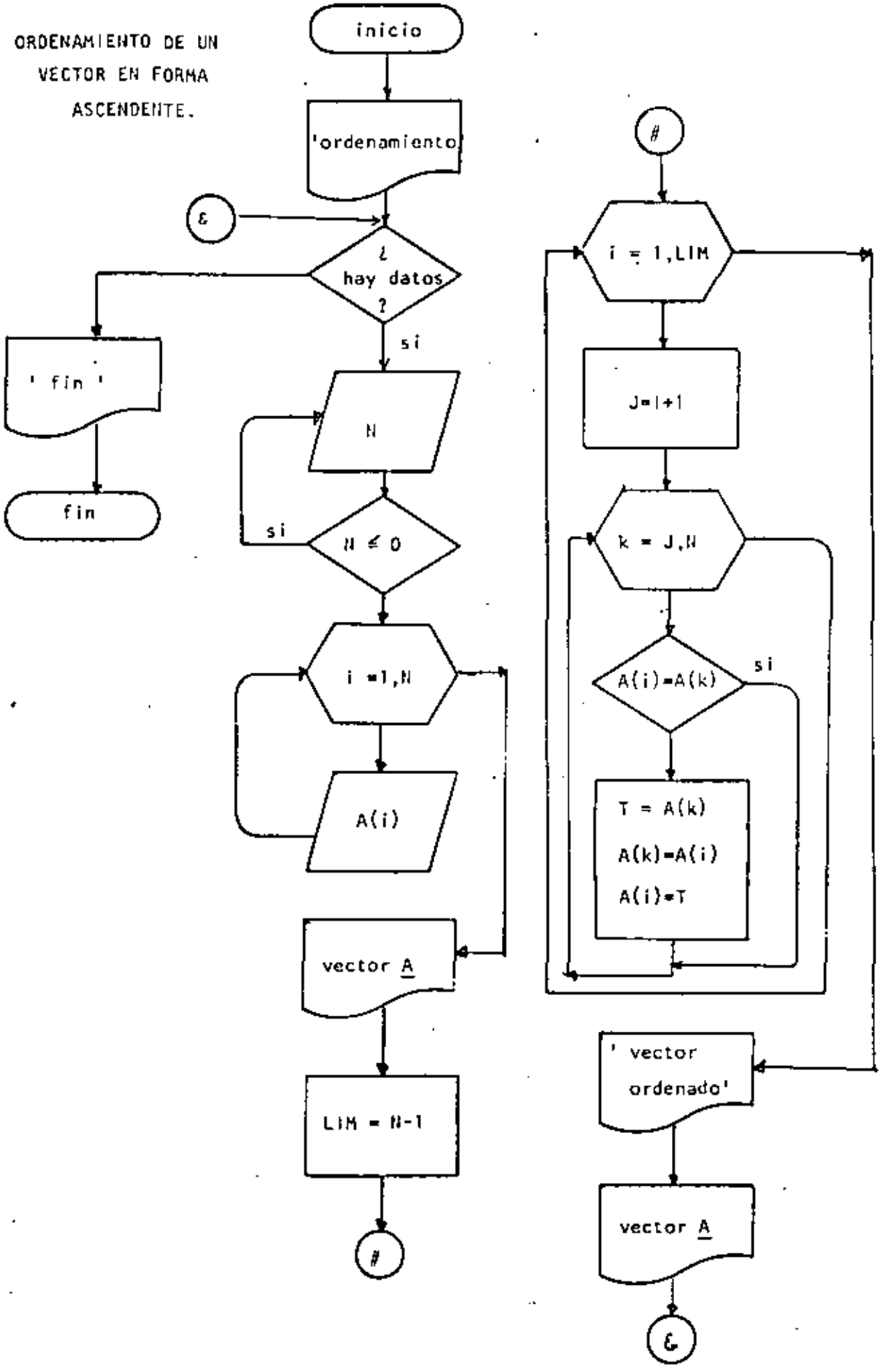

inicio
NUMERO DE BILLETES
N$
N$
&
¿ hay datos ?
si
' fin '
fin
inicializa las vars.
NUM
NUM
si
NUM<10000
DMIL= NUM / 10000
NUM=NUM-DMIL* 10000
NUM < 5000
si
CMIL = 1
#
#
NUM<1000
MIL = MIL +1
NUM=NUM-1000
si
NUM <500
QUIN = 1
NUM=NUM-500
si
NUM <100
SIEN= NUM / 100
NUM=NUM-SIEN * 100
si
NUM <50
C50 = 1
NUM= NUM-50
si
NUM <20
VEIN=VEIN+1
NUM = NUM-20
si
NUM <10
DIEZ = 1
NUM=NUM-10
si
NUM <5
CINCO = 1
NUM = NUM-5
PESO=NUM
Rectifica el total
imprime las
&

```
Ready
LIST
10  REM----TRECE-----
20 REM CALCULO DEL NUMERO DE BILLETES
30 REM
40 REM SE CONSIDERA QUE EXISTEN BILLETES DE DIEZ MIL Y DE CINCO MIL PESOS
50 REM
60 INPUT "NOMBRE DEL CLIENTE";N$
70 PRINT
80 PRINT "HOLA ";N$, "ESPERO QUE ESTE USTED BIEN"
90 PRINT
100 'DOWHILE HAYA DATOS
110 INPUT "HAY DATOS (SI O NO)";S$ : IF S$ <> "SI" THEN 820
120     CLS:'INICIALIZA VARIABLES
130 REM CIEN SE ESCRIBE CON 'C' PERO POR RESTRICCIONES DEL LENGUAJE LO
140 REM ESCRIBIRE CON 'S' PARA QUE NO SE CONFUNDA CON CINCO.
150     READ DMIL,CMIL,MIL,QUIN,SIEN,C50,VEIN,DIEZ,CINCO,PESOS
160     DATA 0,0,0,0,0,0,0,0,0,0
170     INPUT" DAME EL MONTO DEL CHEQUE, SIN CENTAVOS";NUM
180     PRINT USING "CANTIDAD A PAGAR     ###,#########. ";NUM
190     ' CANTIDADES MAYORES DE DIEZ MIL
200     IF NUM < 10000 THEN 240
210             DMIL = FIX( NUM/10000)
220             NUM=NUM-10000*DMIL
230     ' RESTANTE MAYOR QUE 5000
240     IF NUM < 5000 THEN 270
250             CMIL=1
260             NUM=NUM-5000
270     ' DETERMINACION DEL NUMERO DE BILLETES DE MIL
280     NUM%=NUM
290     IF NUM% < 1000 THEN 330
300             MIL=MIL+1
310             NUM%=NUM%-1000
320             GOTO 290
330     ' REVISAR SI ES NECESARIO DAR UN BILLETE DE 500
340     IF NUM% < 500 THEN 380
350             QUIN=1
360             NUM%=NUM%-500
370     ' DETERMINACION DE LOS BILLETES DE 100
380     IF NUM% < 100 THEN 420
390             SIEN=FIX(NUM%/100)
400             NUM%=NUM%-SIEN*100
410     ' REVISAR SI ES NECESARIO DAR UN BILLETE DE 50
420     IF NUM% < 50 THEN 440
430             C50=1:NUM%=NUM%-50
440     ' DETERMINAR EL NUMERO DE BILLETES ( O MONEDAS ) DE 20
450     IF NUM% < 20 THEN 480
460             VEIN=VEIN+1
470             NUM%=NUM%-20 : GOTO 450
480     ' REVISAR SI ES NECESARIO DAR UNA MONEDA DE DIEZ
490     IF NUM% < 10 THEN 530
500             DIEZ=1
510             NUM%=NUM%-10
520     ' DETERMINAR SI ES NECESARIO DAR UNA MONEDA DE CINCO
530     IF NUM% < 5 THEN 570
540             CINCO=1
550             NUM%=NUM%-5
560     ' LO QUE RESTA ES EL NUMERO DE MONEDAS DE UN PESO
570             PESO=NUM%
580     'FIN DEL REPARTO
```

```
590     TTAL=0
600     TTAL=TTAL+ 10000*DMIL
610     IF DMIL> 0 THEN PRINT 'BILLETES DE DIEZ MIL ';TAB(30);DMIL,TTAL
620     TTAL=TTAL+ 5000 *CMIL
630     IF CMIL>0 OR DMIL>0 THEN PRINT 'BILLETES DE CINCO MIL ';TAB(31);CMIL,TTA
L
640     TTAL=TTAL+ 1000 * MIL
650     PRINT 'BILLETES DE MIL ';TAB(31);MIL,TTAL
660     TTAL=TTAL+  500 *QUIN
670     PRINT 'BILLETES DE QUINIENTOS ';TAB(31);QUIN,TTAL
680     TTAL=TTAL+  100 *SIEN
690     PRINT 'BILLETES DE CIEN 'TAB(31);;SIEN,TTAL
700     TTAL=TTAL+   50 *C50
710     PRINT 'BILLETES DE CINCUENTA ';TAB(31);C50,TTAL
720     TTAL=TTAL+   20 *VEIN
730     PRINT 'MONEDAS  DE VEINTE ';TAB(31);VEIN,TTAL
740     TTAL=TTAL+   10 *DIEZ
750     PRINT 'MONEDAS  DE DIEZ   ';TAB(31);DIEZ,TTAL
760     TTAL=TTAL+    5 *CINCO
770     PRINT 'MONEDAS  DE CINCO  ';TAB(31);CINCO,TTAL
780     TTAL=TTAL+    1 *PESO
790     PRINT 'MONEDAS  DE UNO    ';TAB(31);PESO,TTAL
800     RESTORE: GOTO 100
810 'ENDDO
820 PRINT
830 PRINT 'ADIOS ';N$
840 END
Ready
>RUN
NOMBRE DEL CLIENTE? JORGE ONTIVEROS

HOLA JORGE ONTIVEROS            ESPERO QUE ESTE USTED BIEN

HAY DATOS (SI O NO)? SI
DAME EL MONTO DEL CHEQUE, SIN CENTAVOS? 19347
CANTIDAD A PAGAR          19,347.
BILLETES DE DIEZ MIL          1                 10000
BILLETES DE CINCO MIL         1                 15000
BILLETES DE MIL               4                 19000
BILLETES DE QUINIENTOS        0                 19000
BILLETES DE CIEN              3                 19300
BILLETES DE CINCUENTA         0                 19300
MONEDAS  DE VEINTE            2                 19340
MONEDAS  DE DIEZ              0                 19340
MONEDAS  DE CINCO             1                 19345
MONEDAS  DE UNO               2                 19347
HAY DATOS (SI O NO)? SI
DAME EL MONTO DEL CHEQUE, SIN CENTAVOS? 6700
CANTIDAD A PAGAR           6,700.
BILLETES DE CINCO MIL         1                  5000
BILLETES DE MIL               1                  6000
BILLETES DE QUINIENTOS        1                  6500
BILLETES DE CIEN              2                  6700
BILLETES DE CINCUENTA         0                  6700
MONEDAS  DE VEINTE            0                  6700
MONEDAS  DE DIEZ              0                  6700
MONEDAS  DE CINCO             0                  6700
MONEDAS  DE UNO               0                  6700
HAY DATOS (SI O NO)? NO

ADIOS JORGE ONTIVEROS
```

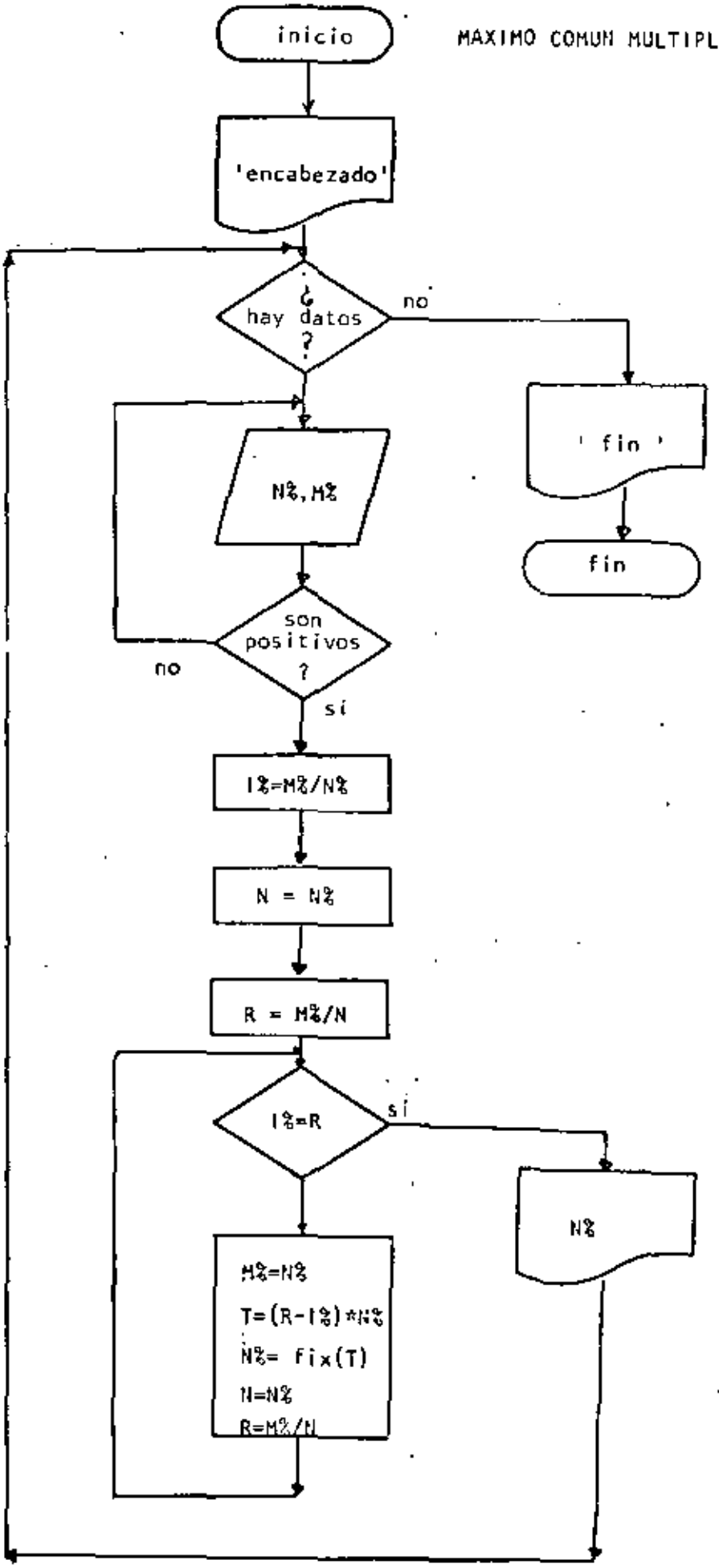
```
Ready
>RUN
NOMBRE DEL CLIENTE? ALGUIEN

HOLA ALGUIEN    ESPERO QUE ESTE USTED BIEN

HAY DATOS (SI O NO)? SI
DAME EL MONTO DEL CHEQUE, SIN CENTAVOS? 123456
CANTIDAD A PAGAR            123,456.
BILLETES DE DIEZ MIL            12              120000
BILLETES DE CINCO MIL            0              120000
BILLETES DE MIL                  3              123000
BILLETES DE QUINIENTOS           0              123000
BILLETES DE CIEN                 4              123400
BILLETES DE CINCUENTA            1              123450
MONEDAS  DE VEINTE               0              123450
MONEDAS  DE DIEZ                 0              123450
MONEDAS  DE CINCO                1              123455
MONEDAS  DE UNO                  1              123456
HAY DATOS (SI O NO)? NO

ADIOS ALGUIEN
Ready
```

ORDENAMIENTO DE UN VECTOR EN FORMA ASCENDENTE.



```
Ready
>LIST
10 REM----CATORCE----
20 REM ORDENAMIENTO ASCENDENTE DE UN VECTOR
30 REM
35 PRINT " PROGRAMA PARA ORDENAR UN VECTOR"
40 'DOWHILE HAYA DATOS
50  INPUT " HAY DATOS (SI O NO)";S$: IF S$ <> "SI" THEN 330
60       INPUT "DAME EL NUMERO DE ELEMENTOS DEL VECTOR";N
70       IF N <= 0 THEN 60
80       DIM A(N)
90       PRINT "DAME LOS ELEMENTOS DEL VECTOR"
100      FOR I=1 TO N
110              INPUT A(I)
120      NEXT I
130      CLS
140      PRINT" TUS ";N; " DATOS SON:"
150      FOR I=1 TO N: PRINT A(I);:NEXT I
160      'SE PROCEDE A ORDENAR EL VECTOR
170      LIM=N-1
180      FOR I=1 TO LIM
190              J=I+1
200              ' SE ASUME QUE A(I) ES EL MENOR
210              FOR K=J TO N
220                      IF A(I) <= A(K) THEN260
230                              ' A(I) FUE > QUE A(K)
240                              T=A(K):A(K)=A(I):A(I)=T
250                              ' SE INTERCAMSIARON
260              NEXT K
270      NEXT I
280      PRINT:PRINT
290      PRINT "VECTOR ORDENADO":PRINT
300      FOR I=1 TO N: PRINT A(I);:NEXT I
305      PRINT
310      GOTO 50
320 'ENDDO
330 PRINT " FIN DEL ORDENAMIENTO"
340 END
Ready
>RUN
 PROGRAMA PARA ORDENAR UN VECTOR
 HAY DATOS (SI O NO)? SI
DAME EL NUMERO DE ELEMENTOS DEL VECTOR? 5
DAME LOS ELEMENTOS DEL VECTOR
? 7
? 9
? 0
? 3
? 5
TUS  5  DATOS SON:
 7   9   0   3   5

VECTOR ORDENADO

 0   3   5   7   9
 HAY DATOS (SI O NO)? NO
 FIN DEL ORDENAMIENTO
Ready
>
```

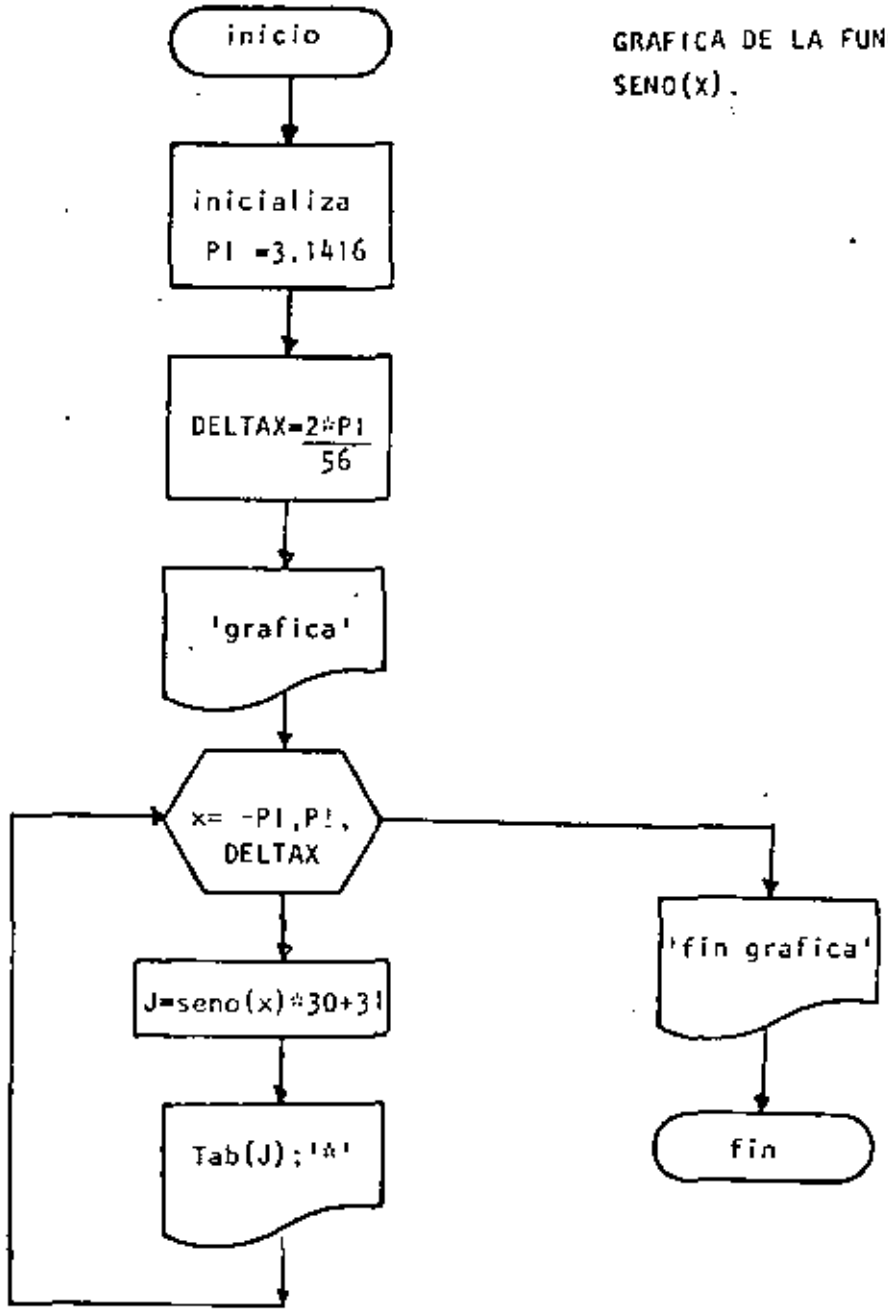
MAXIMO COMUN MULTIPLO



```
1 REM -----QUINCE-----
10 PRINT " PROGRAMA PARA OBTENER EL MAXIMO COMUN DIVISOR DE DOS NUMEROS"
20 PRINT "                USANDO EL ALGORITMO DE EUCLIDES"
30 PRINT
40 REM
50 REM  MAXIMO COMUN MULTIPLO DEL ALGORITMO DE EUCLIDES
60 REM
70 'DOWHILE HAYA DATOS
80  INPUT "HAY DATOS (SI O NO)";S$:IF S$ <> "SI" THEN 290
90       INPUT "DAME LOS NUMEROS PARA BUSCAR EL M.C.D.";NX,MX
100       IF NX <= 0 OR MX <= 0 THEN 90
110       ' SE CALCULA EL RESIDUO
120       IX=MX/NX
130       N=NX
140       R=MX/N
150       'DOWHILE IX <> R
160       IF IX = R THEN 240
170                 MX=NX
180                 T=(R-IX)*NX
190                 NX=FIX(T)
200                 IX=MX/NX
210                 N=NX
220                 R=MX/N
230                 GOTO 160
240       'ENDDO
250       'N REPRESENTA EL MAXIMO COMUN DIVISOR
260       PRINT "EL MAXIMO COMUN DIVISOR ES : ";NX
270       GOTO 80
280 'ENDDO
290 PRINT " FIN DEL MAXIMO COMUN DIVISOR"
```

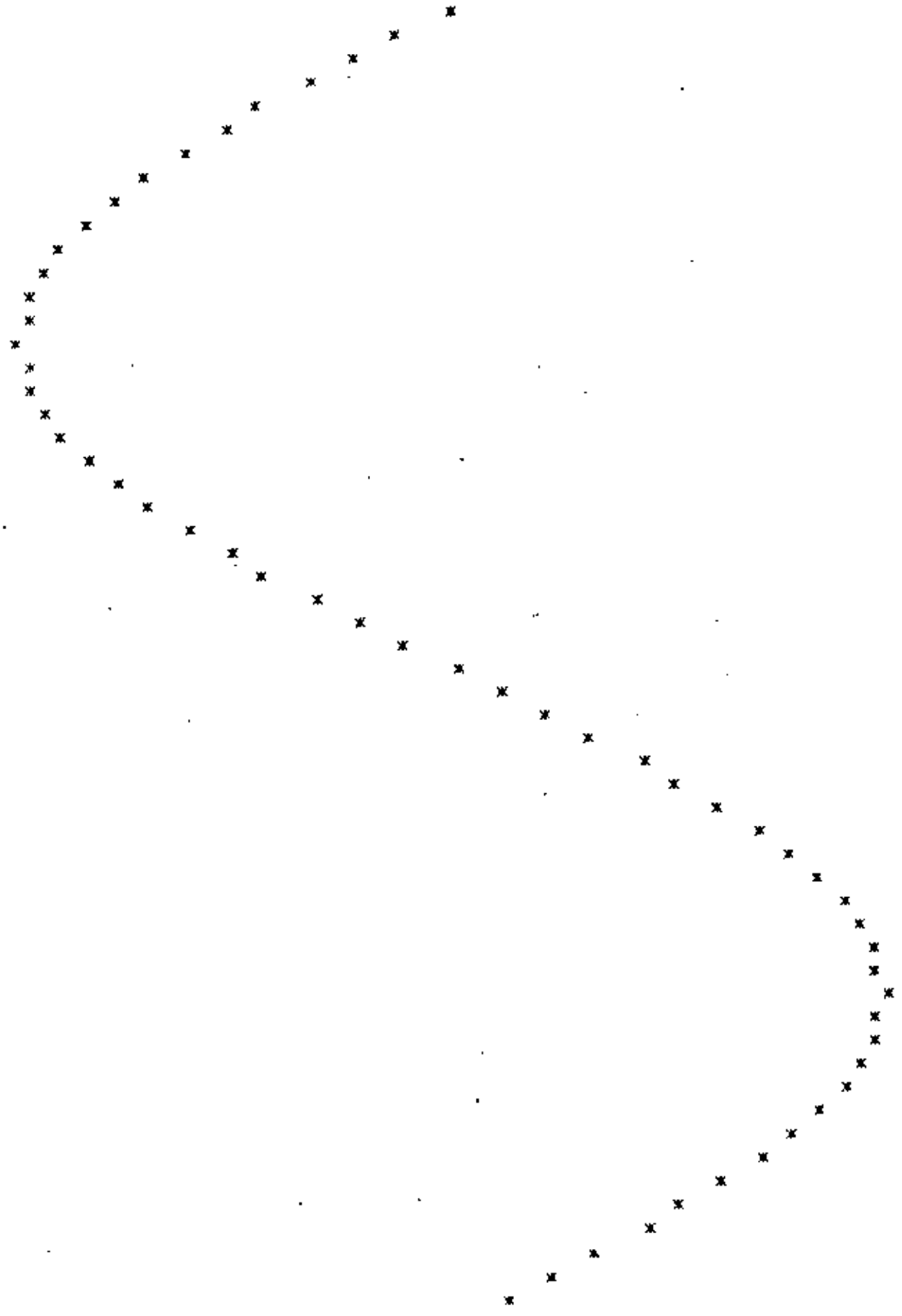
```
>RUN
 PROGRAMA PARA OBTENER EL MAXIMO COMUN DIVISOR DE DOS NUMEROS
               USANDO EL ALGORITMO DE EUCLIDES

HAY DATOS (SI O NO)? SI
DAME LOS NUMEROS PARA BUSCAR EL M.C.D.? 5,8
EL MAXIMO COMUN DIVISOR ES :  1
HAY DATOS (SI O NO)? SI
DAME LOS NUMEROS PARA BUSCAR EL M.C.D.? 13,7
EL MAXIMO COMUN DIVISOR ES :  1
HAY DATOS (SI O NO)? SI
DAME LOS NUMEROS PARA BUSCAR EL M.C.D.? 4,6
EL MAXIMO COMUN DIVISOR ES :  2
HAY DATOS (SI O NO)? SI
DAME LOS NUMEROS PARA BUSCAR EL M.C.D.? 20,25
EL MAXIMO COMUN DIVISOR ES :  5
HAY DATOS (SI O NO)? NO
 FIN DEL MAXIMO COMUN DIVISOR
Ready
```

GRAFICA DE LA FUNCION SENO(X).

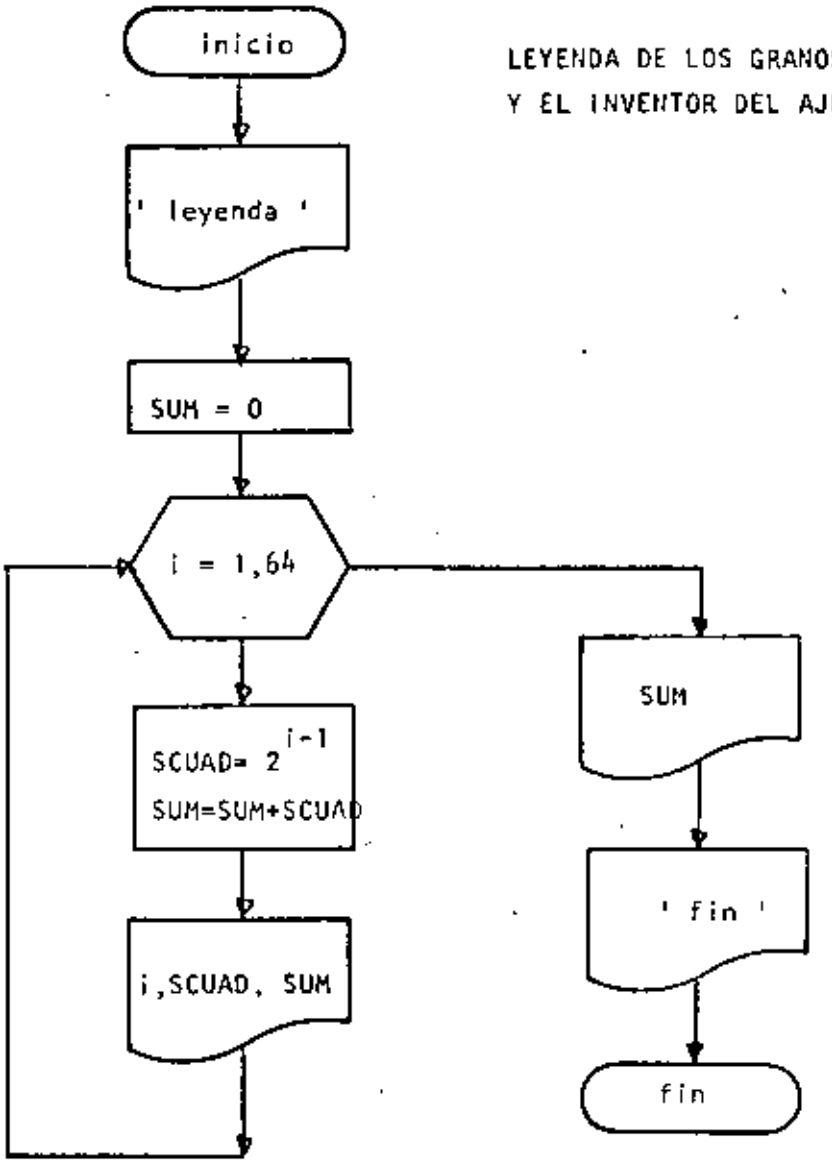



```
10 REM----DIECISEIS----
20 REM GRAFICA DEL SENO
30 REM
40 READ PI: DATA 3.1416
50 DELTAX=2*PI/56
60 PRINT 'GRAFICA DEL SENO'
70 PRINT
80 FOR X=-PI TO PI STEP DELTAX
90       J=SIN(X)*30+31
100      ' LA PANTALLA ES DE 64 POSICIONES
110       PRINT  TAB(J);'*'
120 NEXT X
130 PRINT 'FIN DE LA GRAFICA'
140 END
```

GRAFICA DEL SENO



FIN DE LA GRAFICA



LEYENDA DE LOS GRANOS DE MAIZ
Y EL INVENTOR DEL AJEDREZ.

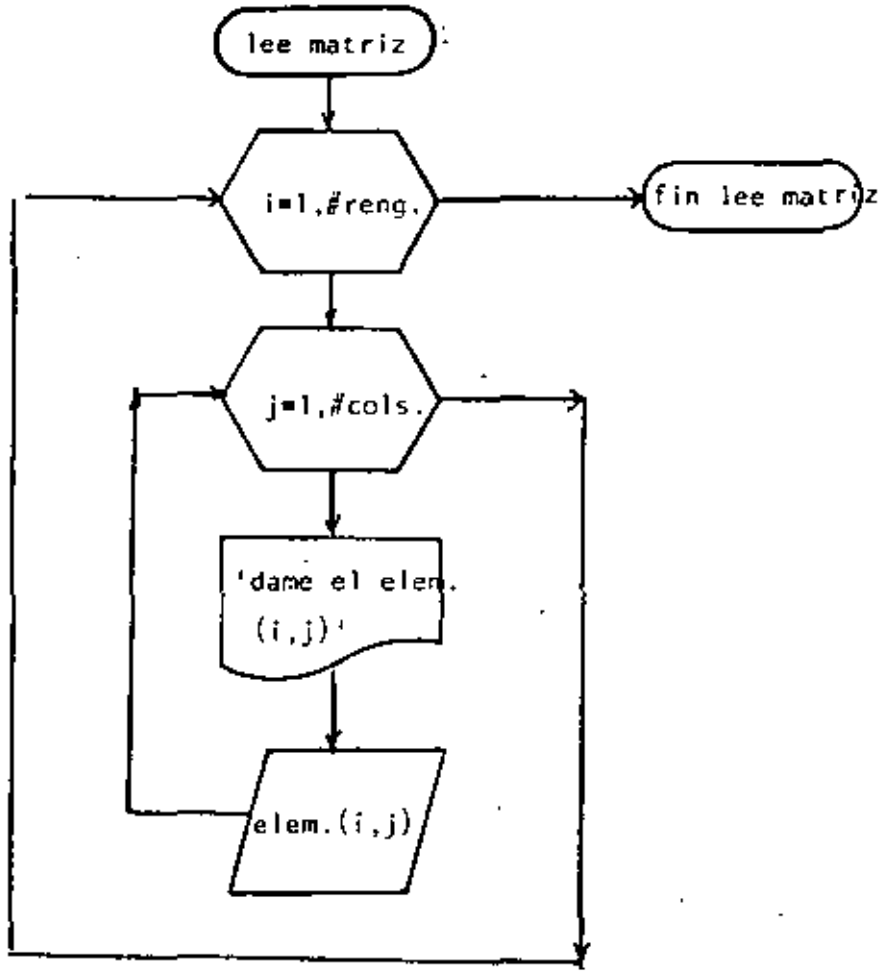
```
10 REM----DIECISIETE----
20 REM
30 REM ESTE PROGRAMA CALCULA EL NUMERO DE GRANOS DE MAIZ QUE COBRO EL INVENTOR
40 REM DEL JUEGO DE AJEDREZ
50 DEFDBL S
55 PRINT: PRINT "LEYENDA DE LOS GRANOS DE MAIZ Y EL AJEDREZ":PRINT
60 SUM=0
65 PRINT "CASILLA     GRANOS QUE LE CORRESPONDEN        TOTAL"
70 FOR I=1 TO 64
80       SCUAD=2↑(I-1)
90       SUM=SUM+SCUAD
100      PRINT TAB(3);I,TAB(10);SCUAD;TAB(45);SUM
110 NEXT I
  5 PRINT:PRINT "TOTAL DE GRANOS";SUM:PRINT
 20 PRINT
130 PRINT"FIN DEL GRANERO"
140 END
```

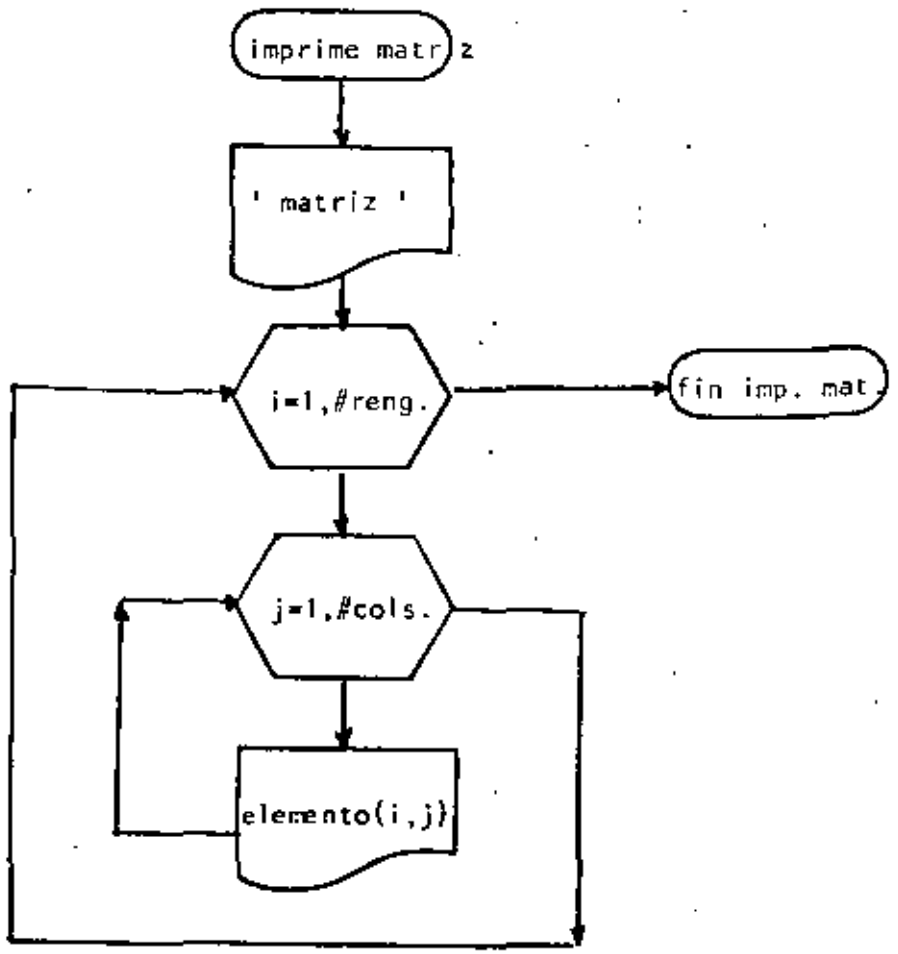
Ready
>RUN

LEYENDA DE LOS GRANOS DE MAIZ Y EL AJEDREZ

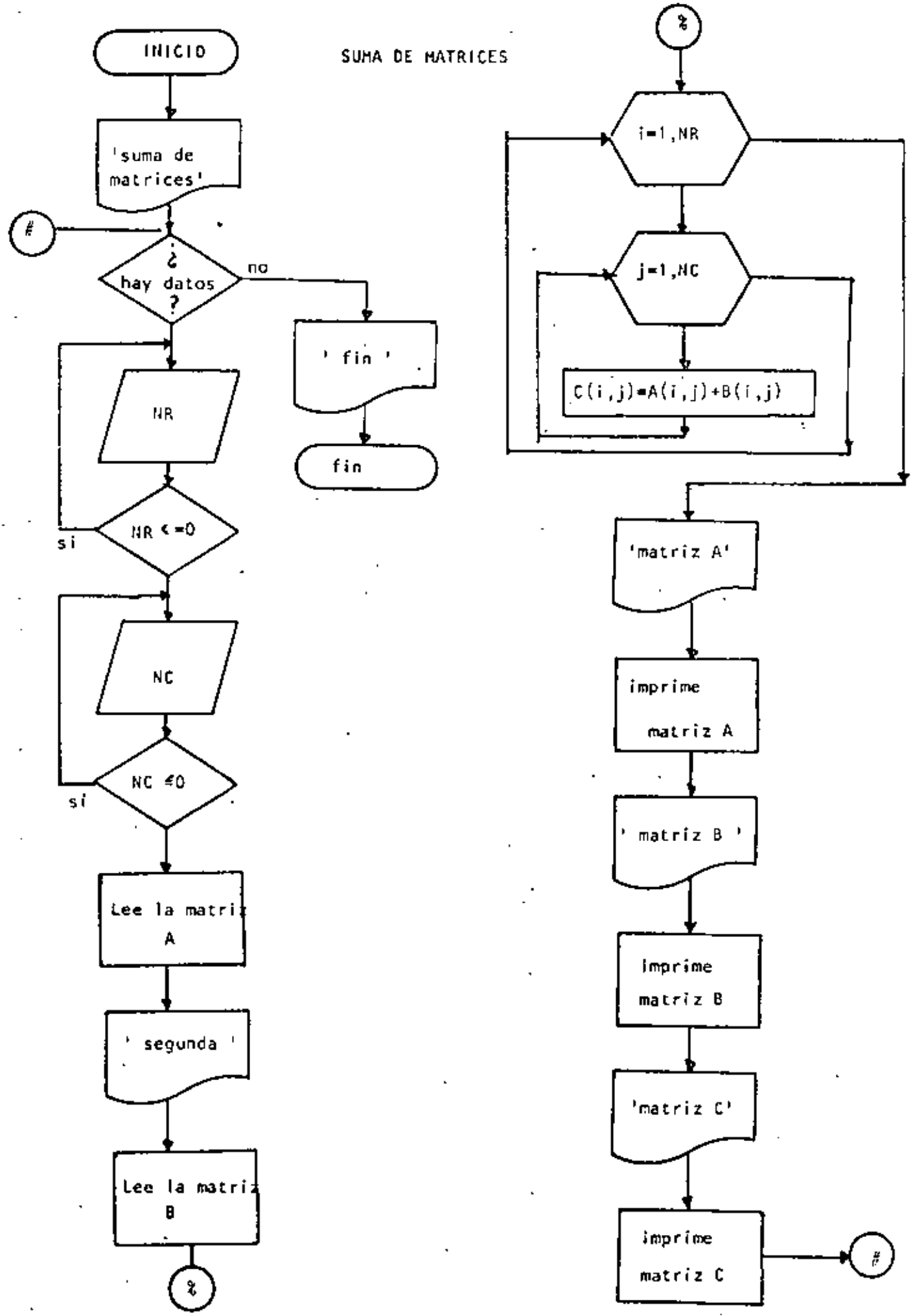
CASILLA	GRANOS QUE LE CORRESPONDEN	TOTAL
1	1	1
2	2	3
3	4	7
4	8	15
5	16	31
6	32	63
7	64	127
8	128	255
9	256	511
10	512	1023
11	1024	2047
12	2040	4095
13	4096	8191
14	8192.005859375	16383.005859375
15	16384	32767.005859375
16	32768.0234375	65535.029296875
17	65536	131071.029296875
18	131072	262143.029296875
19	262144	524287.029296875
20	524288	1048575.029296875
21	1048576	2097151.029296875
22	2097152	4194303.029296875
23	4194304	8388607.029296875
24	8388608	16777215.02929688
25	16777216	33554431.02929688
26	33554432	67108863.02929688
27	67108952	134217815.0292969
28	134217552	268435367.0292969
29	268435456	536870823.0292969
30	536870912	1073741735.029297
31	1073743232	2147484967.029297
32	2147480832	4294965799.029297
33	4294967296	8589933095.029297
34	8589934592	17179867687.0293
35	17179869184	34359736871.0293
36	34359738368	68719475239.0293
37	68719476736	137438951975.0293
38	137438953472	274877905447.0293
39	274877906944	549755812391.0293
40	549755813888	1099511626279.029
41	1099511627776	2199023254055.029
42	2199023255552	4398046509607.029
43	4398046511104	8796093020711.029
44	8796093022208	17592186042919.03
45	17592186044416	35184372087335.03
46	35184372088832	70368744176167.03
47	70368744177664	140737488353831
48	140737488355328	281474976709159
49	281474976710656	562949953419815
50	562948477026304	1125878430446119
51	1125899906842624	2251798337288743

```
 52              2251799813685248              4503598150973991
 53              4503611438530560              9007209589504551
 54              9007199254740992              1.8014408844245540+16
 55              1.8014351264841730+16         3.6028760109087270+16
 56              3.6028797018963970+16         7.2057557128051240+16
 57              7.2057594037927940+16         1.4411515116597920+17
 58              1.4411481011873380+17         2.8822996128471300+17
 59              2.8822037615171170+17         5.7646033743642480+17
 60              5.7646075230342350+17         1.1529210897398480+18
 61              1.1529245282638230+18         2.3058456180036720+18
 62              2.3058430092136940+18         4.6116886272173660+18
 63              4.6116739237994820+18         9.2233625510168480+18
 64              9.2233720368547760+18         1.8446734587871620+19

TOTAL DE GRANOS 1.8446734587871620+19


FIN DEL GRANERO
Ready
```


lee matriz
i=1,#reng.
fin lee matriz
j=1,#cols.
'dame el elem.
(i,j)'
elem.(i,j)


imprime matriz
' matriz '
i=1,#reng.
fin imp. mat.
j=1,#cols.
elemento(i,j)

SUMA DE MATRICES

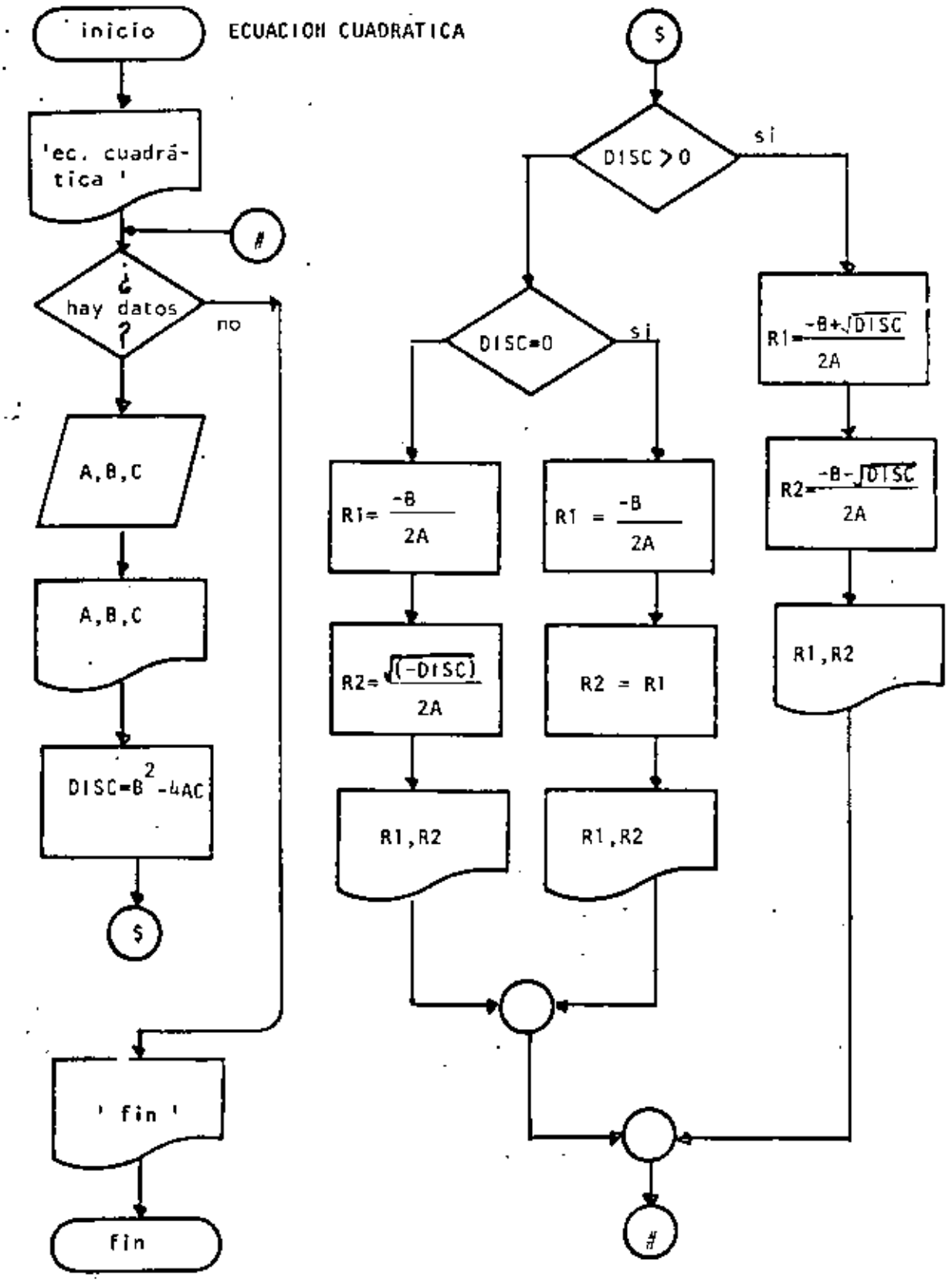


```
10 REM-----DIECIOCHO----
20 REM SUMA DE MATRICES DE MAXIMO 10 POR 10
30 REM
40 PRINT "SUMA DE MATRICES DE MAXIMO 10 POR 10":PRINT
50 'DOWHILE HAYA DATOS
60 INPUT "HAY DATOS(SI O NO)";S$
70 IF S$ <> "SI" THEN 630
75       PRINT"DATOS DE LA PRIMERA MATRIZ":PRINT
80       INPUT "NUMERO DE RENGLONES";NR : IF NR <= 0 THEN 80
90       INPUT "NUMERO DE COLUMNAS ";NC : IF NC <= 0 THEN 90
100      CLS:REM LECTURA DE LA MATRIZ A
110      FOR I=1 TO NR
120         FOR J=1 TO NC
130              PRINT @ 720,  "DAME EL ELEMENTO(";I;",";J;")";
140              INPUT A(I,J)
150         NEXT J
160      NEXT I
170      PRINT
180      PRINT "DATOS DE LA SEGUNDA MATRIZ"
190      PRINT
200      FOR I=1 TO NR
210         FOR J=1 TO NC
220              PRINT @960, "DAME EL ELEMENTO(";I;",";J;")";
230              INPUT B(I,J)
240         NEXT J
250      NEXT I
260      PRINT
270      REM
280      REM SUMA DE LAS DOS MATRICES
290      REM
300      FOR I=1 TO NC
310              FOR J=1 TO NC
320                       C(I,J)=A(I,J)+B(I,J)
330              NEXT J
340      NEXT I
350      REM
360      REM SALIDA DE RESULTADOS
370      REM
380      CLS
390      PRINT:PRINT  "MATRIZ A":PRINT
400      FOR I=1 TO NR
410              FOR J=1 TO NC
420                       PRINT USING "#####.#";A(I,J);
430              NEXT J
440              PRINT
450      NEXT I
460      PRINT:PRINT "MATRIZ B":PRINT
470      FOR I=1 TO NR
480              FOR J=1 TO NC
490                       PRINT USING "#####.#";B(I,J);
500              NEXT J
510              PRINT
520      NEXT I
530      PRINT:PRINT "MATRIZ RESULTADO":PRINT
540      FOR I=1 TO NR
550              FOR J=1 TO NC
560                       PRINT USING "######.#";C(I,J);
570              NEXT J
580              PRINT
590      NEXT I
```

```
600     PRINT'
610 GOTO 50
620 'ENDDO
630 PRINT "FIN DE LA SUMA DE MATRICES"
640 END

>RUN
SUMA DE MATRICES DE MAXIMO 10 POR 10

HAY DATOS(SI O NO)? SI
DATOS DE LA PRIMERA MATRIZ

NUMERO DE RENGLONES? 3
NUMERO DE COLUMNAS ? 4
AME EL ELEMENTO( 1 , 1 )? 4
DAME EL ELEMENTO( 1 , 2 )?
DAME EL ELEMENTO( 1 , 3 )?
DAME EL ELEMENTO( 1 , 4 )?
DAME EL ELEMENTO( 2 , 1 )? 3
DAME EL ELEMENTO( 2 , 2 )?
DAME EL ELEMENTO( 2 , 3 )? 8
DAME EL ELEMENTO( 2 , 4 )?
DAME EL ELEMENTO( 3 , 1 )?
DAME EL ELEMENTO( 3 , 2 )? 2
DAME EL ELEMENTO( 3 , 3 )? 4
DAME EL ELEMENTO( 3 , 4 )? 7

DATOS DE LA SEGUNDA MATRIZ

DAME EL ELEMENTO( 1 , 1 )? 2
DAME EL ELEMENTO( 1 , 2 )? 5
DAME EL ELEMENTO( 1 , 3 )? 6
DAME EL ELEMENTO( 1 , 4 )? 3
DAME EL ELEMENTO( 2 , 1 )? 0
DAME EL ELEMENTO( 2 , 2 )? 1
DAME EL ELEMENTO( 2 , 3 )? 4
DAME EL ELEMENTO( 2 , 4 )? 7
DAME EL ELEMENTO( 3 , 1 )? 8
DAME EL ELEMENTO( 3 , 2 )? 9
DAME EL ELEMENTO( 3 , 3 )? 6
DAME EL ELEMENTO( 3 , 4 )? 3

MATRIZ A

     4.0     0.0     0.0     0.0
     3.0     0.0     8.0     0.0
     0.0     2.0     4.0     7.0

MATRIZ B

     2.0     5.0     6.0     3.0
     0.0     1.0     4.0     7.0
     8.0     9.0     6.0     3.0

MATRIZ RESULTADO

     6.0     5.0     6.0     3.0
     3.0     1.0    12.0     7.0
     8.0    11.0    10.0    10.0

HAY DATOS(SI O NO)? NO
FIN DE LA SUMA DE MATRICES
```

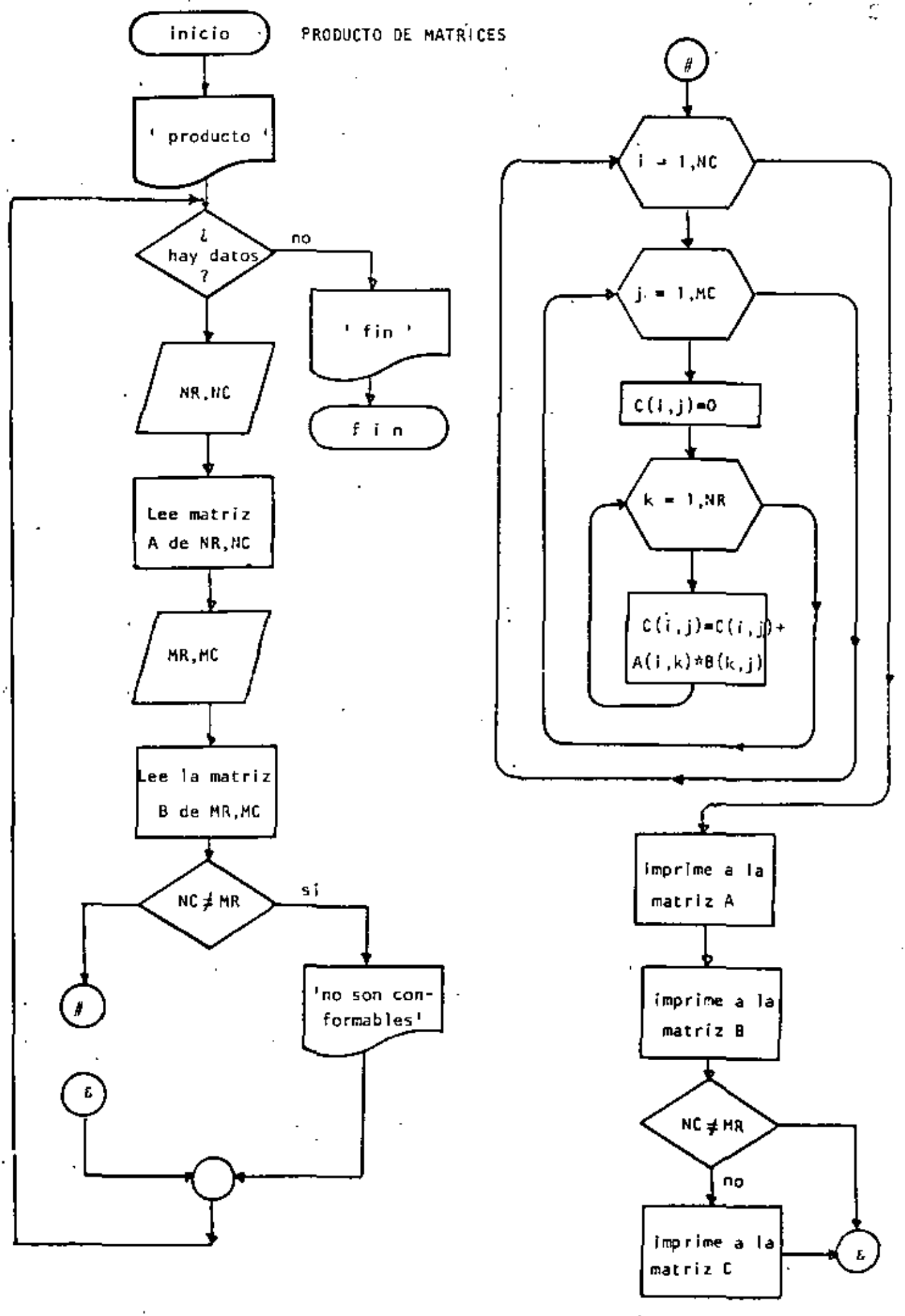
```
Ready
>RUN
SOLUCION DE LA ECUACION CUADRATICA

 HAY DATOS(SI O NO)? SI
 DAME LOS COEFICIENTES A,B Y C? 1,5,3
 LA ECUACION A RESOLVER ES
 1  X↑2 + 5  X + 3  = 0

RAICES REALES DIFERENTES
RAIZ 1 =-.697225                    RAIZ 2 =-4.30278

 HAY DATOS(SI O NO)? SI
 DAME LOS COEFICIENTES A,B Y C? 1,6,2
 LA ECUACION A RESOLVER ES
 1  X↑2 + 6  X + 2  = 0

RAICES REALES DIFERENTES
RAIZ 1 =-.354249                    RAIZ 2 =-5.64575

 HAY DATOS(SI O NO)? SI
 DAME LOS COEFICIENTES A,B Y C? 1,0,4
 LA ECUACION A RESOLVER ES
 1  X↑2 + 0  X + 4  = 0

RAICES COMPLEJAS CONJUGADAS
 RAIZ 1 = ( 0 , 2 )
 RAIZ 2  =( 0 ,-2 )

 HAY DATOS(SI O NO)? SI
 DAME LOS COEFICIENTES A,B Y C? 0,8,9
 LA ECUACION A RESOLVER ES
 0  X↑2 + 8  X + 9  = 0

?/0 Error in 150
Ready
>RUN
SOLUCION DE LA ECUACION CUADRATICA

 HAY DATOS(SI O NO)? SI
 DAME LOS COEFICIENTES A,B Y C? 3,7,5
 LA ECUACION A RESOLVER ES
 3  X↑2 + 7  X + 5  = 0

RAICES COMPLEJAS CONJUGADAS
 RAIZ 1 = (-1.16667 , .552771 )
 RAIZ 2  =(-1.16667 ,-.552771 )

 HAY DATOS(SI O NO)? NO
FIN DE ECUACIONES CUADRATICAS
Ready
```

Esta página está a propósito en este lugar, el diagrama de flujo y el listado del programa están en las páginas siguientes.


inicio
ECUACION CUADRATICA
'ec. cuadrática '
#
¿ hay datos ?
no
A,B,C
A,B,C
DISC=B^2-4AC
$
' fin '
fin
$
DISC > 0
si
DISC=0
si
R1= -B / 2A
R2= √(-DISC) / 2A
R1 = -B / 2A
R2 = R1
R1= (-B+√DISC) / 2A
R2= (-B-√DISC) / 2A
R1,R2
R1,R2
R1,R2
#

```
10 REM----DIECINUEVE----
20 REM SOLUCION DE ECUACIONES CUADRATICAS
30 REM
40 PRINT "SOLUCION DE LA ECUACION CUADRATICA":PRINT
50 'DOWHILE HAYA DATOS
60  INPUT" HAY DATOS(SI O NO)";S$: IF S$ <> "SI" THEN 440
70       INPUT" DAME LOS COEFICIENTES A,B Y C";A,B,C
80       PRINT " LA ECUACION A RESOLVER ES "
90       PRINT A;" X↑2 +";B; " X +";C; " = 0":PRINT
100      DISC=B↑2-4*A*C
110      ' IF RAICES REALES DIFERENTES
120        IF DISC > 0 THEN 140
130                    GOTO 220
140      '  THEN
150             R1= (-B + SQR(DISC))/(2*A)
160             R2= (-B - SQR(DISC))/(2*A)
170             PRINT "RAICES REALES DIFERENTES"
180             PRINT "RAIZ 1 =";R1, "RAIZ 2 =";R2
190             PRINT
200             GOTO 410
210      '  ELSE
220             'IF RAICES REALES IGUALES
230              IF DISC = 0 THEN 250
240                         GOTO 320
250             'THEN
260                     R1 = -B/2/A
270                     R2 = R1
280                     PRINT "RAICES REALES IGUALES"
290                     PRINT "RAIZ 1 =";R1, "RAIZ 2 =";R2
300                     PRINT
310                     GOTO 400
320             'ELSE
330                     'RAICES COMPLEJAS CONJUGADAS
340                     R1 = -B/2/A
350                     R2 = SQR(-DISC)/(2*A)
360                     PRINT "RAICES COMPLEJAS CONJUGADAS"
370                     PRINT " RAIZ 1 = (";R1;",";R2;")"
380                     PRINT " RAIZ 2  =(";R1;",";-R2;")"
390                     PRINT
400             'ENDIF
410     'ENDIF
420 'ENDDO
430 GOTO 50
440 PRINT"FIN DE ECUACIONES CUADRATICAS"
450 END
```

PRODUCTO DE MATRICES

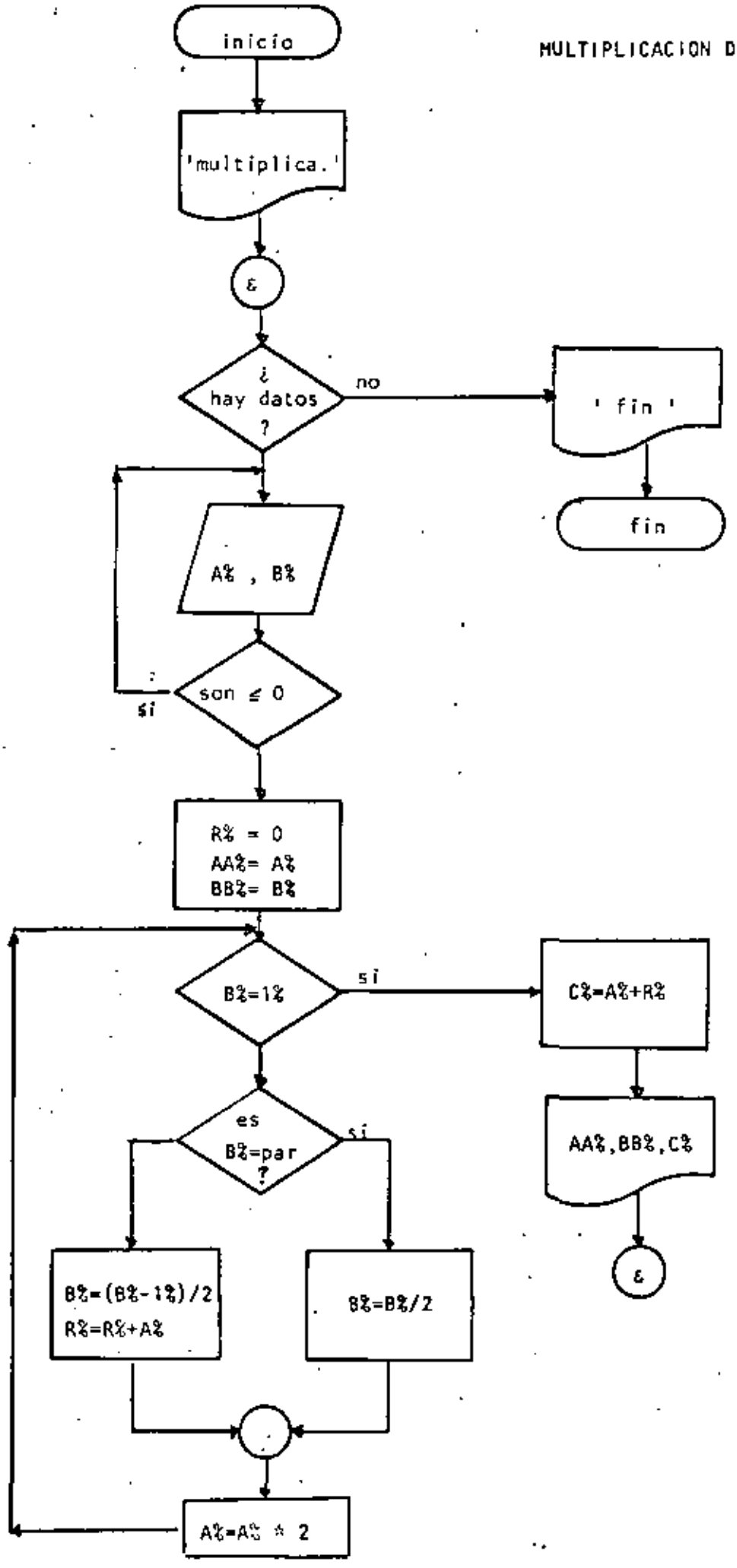


```
10 REM ----VEINTE----
20 REM PRODUCTO DE MATRICES DE MAXIMO 10 POR 10
30 REM
40 PRINT "PRODUCTO DE MATRICES DE MAXIMO 10 POR 10":PRINT
50 'DOWHILE HAYA DATOS
60 INPUT "HAY DATOS(SI O NO)";S$
70 IF S$ <> "SI" THEN 720
80       PRINT "DATOS DE LA PRIMERA MATRIZ":PRINT
90       INPUT "NUMERO DE RENGLONES";NR : IF NR <= 0 THEN 90
100      INPUT "NUMERO DE COLUMNAS ";NC : IF NC <= 0 THEN 100
110      REM LECTURA DE LA MATRIZ A
120      FOR I=1 TO NR
130         FOR J=1 TO NC
140              PRINT "DAME EL ELEMENTO(";I;",";J;")";
150              INPUT A(I,J)
160         NEXT J
170      NEXT I
180      PRINT
190      PRINT "DATOS DE LA SEGUNDA MATRIZ"
200      PRINT
210      INPUT "NUMERO DE RENGLONES";MR: IF MR <= 0 THEN 210
220      INPUT "NUMERO DE COLUMNAS ";MC: IF MC <= 0 THEN 220
230      FOR I=1 TO MR
240         FOR J=1 TO MC
250              PRINT "DAME EL ELEMENTO(";I;",";J;")";
260              INPUT B(I,J)
270         NEXT J
280      NEXT I
290      PRINT
300      REM
310      REM PRODUCTO DE LAS DOS MATRICES
320      REM
330      IF NC <> MR THEN 450: REM NO SON CONFORMABLES
340      FOR I=1 TO NR
350              FOR J=1 TO MC
360                      C(I,J)=0
370                      FOR K = 1 TO MC
380                              C(I,J) = C(I,J) + A(I,K) * B(K,J)
390                      NEXT K
400              NEXT J
410      NEXT I
420      REM
430      REM SALIDA DE RESULTADOS
440      REM
450      CLS: IF NC <> MR THEN PRINT " NO SON CONFORMABLES"
460      PRINT:PRINT  "MATRIZ A":PRINT
470      FOR I=1 TO NR
480              FOR J=1 TO NC
490                      PRINT USING "######.#";A(I,J);
500              NEXT J
510              PRINT
520      NEXT I
530      PRINT:PRINT "MATRIZ B":PRINT
540      FOR I=1 TO MR
550              FOR J=1 TO MC
560                      PRINT USING "######.#";B(I,J);
570              NEXT J
580              PRINT
590      NEXT I
600      PRINT
```

```
610     IF NC <> MR THEN 700
620     PRINT:PRINT "MATRIZ RESULTADO":PRINT
630     FOR I=1 TO NR
640             FOR J=1 TO MC
650                     PRINT USING "#####.#";C(I,J);
660             NEXT J
670             PRINT
680     NEXT I
690     PRINT
700 GOTO 50
710 'ENDDO
720 PRINT "FIN DE LA PRODUCTO DE MATRICES"
730 END
```

```
>RUN
PRODUCTO DE MATRICES DE MAXIMO 10 POR 10

HAY DATOS(SI O NO)? SI
DATOS DE LA PRIMERA MATRIZ

NUMERO DE RENGLONES? 2
NUMERO DE COLUMNAS ? 3
DAME EL ELEMENTO( 1 , 1 )? 2
DAME EL ELEMENTO( 1 , 2 )? 5
DAME EL ELEMENTO( 1 , 3 )? 9
DAME EL ELEMENTO( 2 , 1 )? 2
DAME EL ELEMENTO( 2 , 2 )? 0
DAME EL ELEMENTO( 2 , 3 )? 5

DATOS DE LA SEGUNDA MATRIZ

NUMERO DE RENGLONES? 3
NUMERO DE COLUMNAS ? 4
DAME EL ELEMENTO( 1 , 1 )? 2
DAME EL ELEMENTO( 1 , 2 )? 5
DAME EL ELEMENTO( 1 , 3 )? 9
DAME EL ELEMENTO( 1 , 4 )? 3
DAME EL ELEMENTO( 2 , 1 )? 0
DAME EL ELEMENTO( 2 , 2 )? 1
DAME EL ELEMENTO( 2 , 3 )? 4
DAME EL ELEMENTO( 2 , 4 )? 7
DAME EL ELEMENTO( 3 , 1 )? 5
DAME EL ELEMENTO( 3 , 2 )? 9
DAME EL ELEMENTO( 3 , 3 )? 2
DAME EL ELEMENTO( 3 , 4 )? 2

MATRIZ A

    2.0     5.0     9.0
    2.0     0.0     5.0

MATRIZ B

    2.0     5.0     9.0     3.0
    0.0     1.0     4.0     7.0
    5.0     9.0     2.0     2.0


MATRIZ RESULTADO

   49.0    96.0    56.0    59.0
   29.0    55.0    20.0    16.0

HAY DATOS(SI O NO)? NO
FIN DE LA PRODUCTO DE MATRICES
Ready
```

MULTIPLICACION DE DOS NUMEROS



```
10 REM----VENTIUNO----
20 REM MULTIPLICACION DE DOS NUMEROS UTILIZANDO EXCLUSIVAMENTE
30 REM MULTIPLICACION Y DIVISION ENTRE DOS
40 REM
50 PRINT" MULTIPLICACION DE 2 NUMEROS UTILIZANDO PRODUCTO Y "
60 PRINT" DIVISION ENTRE DOS":PRINT
70 INPUT "HAY DATOS (SI O NO)";S$:IF S$ <> "SI" THEN 320
80 'DOWHILE HAYA DATOS
90      INPUT"DAME LOS VALORES DE A Y B";AX,BX
100     IF AX <=  0  OR BX <=  0 THEN 90
110     RX=0
120     AAX=AX
130     BBX=BX
140     'DOWHILE BX > 1
150     IF BX = 1% THEN 220
160             IF BX = FIX(BX/2)*2 THEN 180
170                                 GOTO 210
180             ' THEN
190                     BX=BX/2
200                     GOTO 240
210             ' ELSE
220                     BX=(BX-1%)/2
230                     RX=RX+AX
240             'ENDIF
250             AX=AX*2
260     GOTO 150
270     'ENDDO
280     CX=AX+RX
290     PRINT "A VALE ";AAX,"B VALE ";BBX,"EL PRODUCTO ES ";CX
300     GOTO 70
310 'ENDDO
320 PRINT
330 PRINT" FIN DE LA MULTIPLICACION"
340 END

Ready

>RUN
 MULTIPLICACION DE 2 NUMEROS UTILIZANDO PRODUCTO Y
 DIVISION ENTRE DOS

HAY DATOS (SI O NO)? SI
DAME LOS VALORES DE A Y B? 6,9
A VALE  6       B VALE  9       EL PRODUCTO ES  54
HAY DATOS (SI O NO)? SI
DAME LOS VALORES DE A Y B? 45,76
A VALE  45      B VALE  76      EL PRODUCTO ES  3420
HAY DATOS (SI O NO)? SI
DAME LOS VALORES DE A Y B? 0,4
DAME LOS VALORES DE A Y B? 1,5
A VALE  1       B VALE  5       EL PRODUCTO ES  5
HAY DATOS (SI O NO)? NO

 FIN DE LA MULTIPLICACION
Ready
```

DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA U.N.A.M.

LENGUAJE DE PROGRAMACION BASIC CON APLICACIONES (PRIMERA PARTE)

SISTEMA DIRECTORIO

M.EN C. JOSE RICARDO CIRIA MERCE

MARZO, 1983

ESPECIFICACIONES

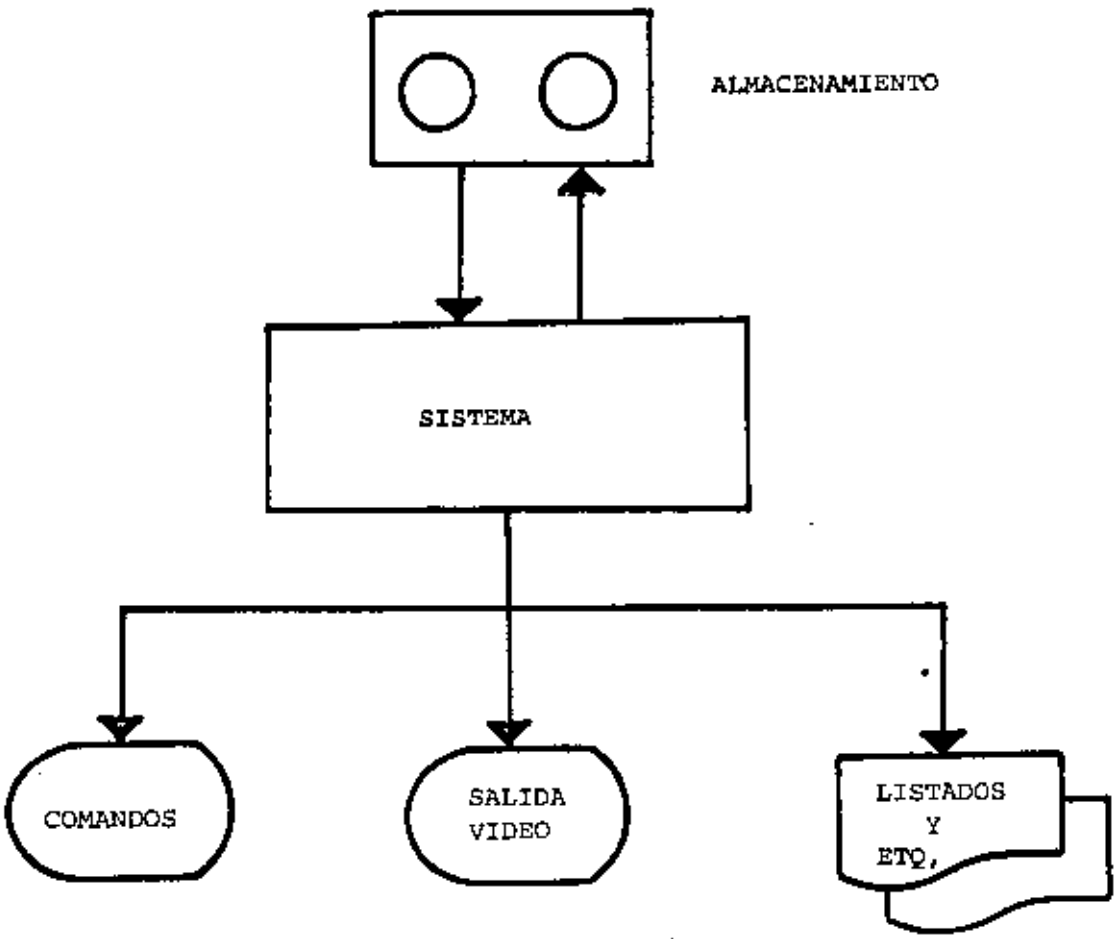
-MANEJAR AUTOMATIZADAMENTE NOMBRES , DOMICILIOS Y DOS SALDOS , TANTO DE CLIENTES COMO DE PRO - VEEDORES , EN UNDIRECTORIO ACTUALIZABLE CON LAS SIGUIENTES CARACTERISTICAS :

I- CAPACIDAD DE MOSTRAR LOS DATOS DE UN CLIENTE O PROVEEDOR EN PARTICULAR , EN LA PANTALLA.

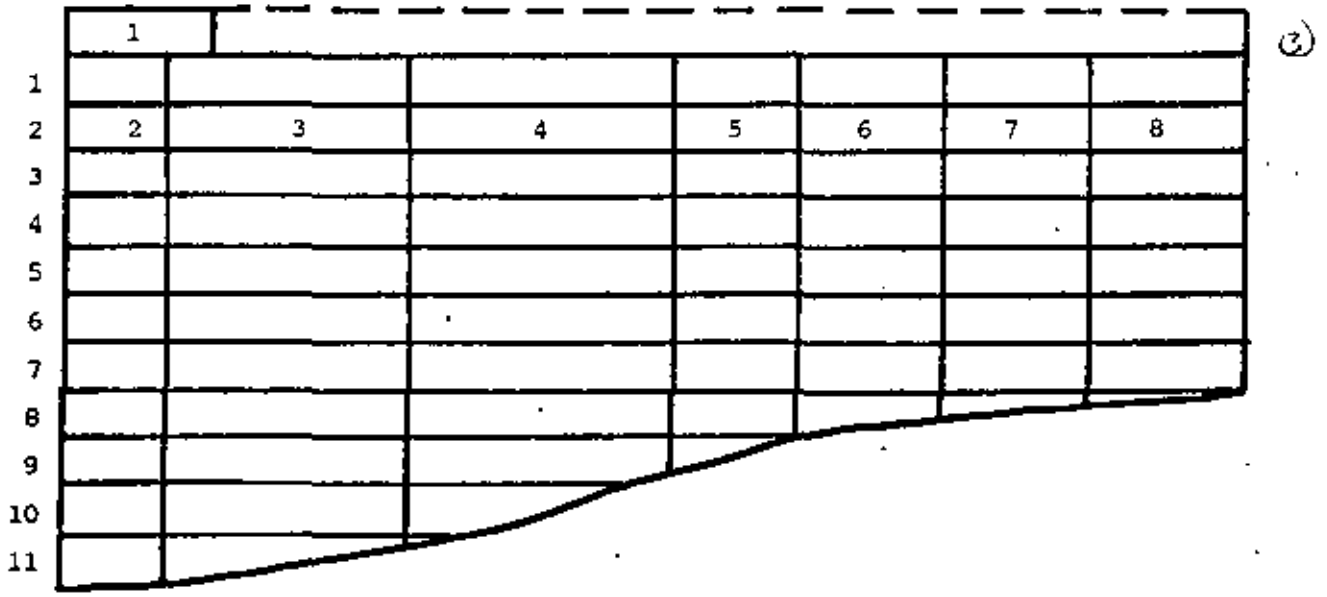
II- CAPACIDAD DE MOSTRAR LOS DATOS DE TODOS LOS CLIENTES O TODOS LOS PROVEEDORES O TODO EL DIRECTORIO EN LA PANTALLA.

III- CAPACIDAD DE LISTAR LOS DATOS DE TODOS LOS CLIENTES O TODOS LOS PROVEEDORES O TODO EL DIRECTORIO.

IV- CAPACIDAD DE IMPRIMIR ETIQUETAS ENGOMADAS PARA CORREO CON LOS DATOS DE TODOS LOS CLIENTES O TODOS LOS PROVEEDORES O TODO EL DIRECTORIO.



" DIAGRAMA DE BLOQUES "



NUMERO	VARIABLE ASOCIADA	TIPO	(LONGITUD)	COMENTARIOS
1	CD	N	—	Número de elementos en el directorio
2	MK$(i)	A	3	llave de acceso; valores admisibles C00 - C99 y P00 - P99
3	MN$(i)	A	32	Nombre
4	MD$(i)	A	40	Dirección
5	DC$(i)	A	15	Colonia
6	MP (i)	N	—	Código Postal
7	M1 (i)	N	—	Saldo 1
8	M2 (i)	N	—	Saldo 2

" ESTRUCTURA DE DATOS "

1.- SALIDA DEL SISTEMA

2.- ALTA

3.- BAJA

4.- CAMBIO

5.- MUESTRA DIRECTORIO (PANTALLA)

6.- LISTA DIRECTORIO

7.- IMPRIME ETIQUETAS

8.- BUSCA

ENTRADA AL SISTEMA: Verificar si es la primera vez.

SALIDA DEL SISTEMA: Verificar si se efectuaron cambios.

ALTA: Verificar : a) si no existe ya la clave
b) si hay espacio en la tabla
c) si es válida la clave a dar de alta.

BAJA,

CAMBIO,

BUSCA :

Verificar si existe la clave.

LISTA:

IMPRIME: Preguntar C, P ó Ambos.

" CONDICIONES DE ACCESO "

CAPACIDAD

-EL SISTEMA ADMITE COMO MAXIMO DIEZ ELEMENTOS (ENTRE CLIENTES Y PROVEEDORES). SI SE DESEA AUMENTAR ESTA CAPACIDAD HABRA QUE DIMENSIONAR LAS VARIABLES (ARREGLOS) QUE APARECEN EN LA FIGURA "ESTRUCTURA DE DATOS" ASI COMO EL VALOR ASIGNADO A LA VARIABLE "MX"

-SI EL CODIGO POSTAL INDICADO PARA ALGUN REGISTRO NO CORRESPONDE AL DISTRITO FEDERAL NO SE IMPRIMIRA (O MOSTRARA EN PANTALLA) DELEGACION ALGUNA. EN SU LUGAR SE IMPRIMIRA (O MOSTRARA EN PANTALLA) UN "STRING" DE DIEZ CARACTERES "BLANCO".

-NINGUN SALDO PUEDE EXCEDER DE $999,999. SI SE DESEA AUMENTAR ESTA CAPACIDAD HABRA QUE MODIFICAR EL VALOR DE LA VARIABLE "F1$" Y , POSIBLEMENTE DECLARAR DE "DOBLE PRECISION" A LAS VARIABLES RELACIONADAS CON SALDOS :

M1(i) , M2(i) , D1 , D2 , T1 y T2

-LOS DATOS RELACIONADOS CON : NOMBRE , DOMICILIO (CALLE Y NUMERO) , Y COLONIA PODRAN TENER UNA LONGITUD DE HASTA 255 CARACTERES , SIN EMBARGO , SE DEBERA TENER CUIDADO DE QUE DICHOS DATOS NO EXCEDAN LA CAPACIDAD DE IMPRESION , SOBRE TODO EN ETIQUETAS.

-LAS CLAVES DE "CLIENTES" SE FORMARAN SIEMPRE DE TRES CARACTERES :
EL PRIMERO SERA LA LETRA "C" Y LOS DOS SIGUIENTES FORMANDO UN NUMERO
ENTRE 00 y 99.
EJEMPLO : C03 , C84 , C00 etc.

-LAS CLAVES DE "PROVEEDORES" SE FORMARAN DE MANERA SIMILAR A LAS
DE CLIENTES , PERO INICIANDO CON LA LETRA "P".
ELEMPLO : P03 , P33 , P00 etc.

-PARA "CARGAR" EL SISTEMA A MEMORIA : PONER LA GRABADORA EN
"PLAY" CON VOLUMEN = 5 Y TECLEAR CLOAD"A".

LISTA DE VARIABLES

C1	si C1=1,se ha alterado el directorio
CA	si CA=1,se desea cambiar un elemento del directorio.
CD	número de elementos válidos en el directorio.
DE	valor numérico de los dos primeros dígitos del Código Postal (Delegación)
DL$	Delegación (nombre)
E1$,EN$	encabezados
F1$	contiene el formato para la impresion de saldos.
ID	si OK=1,contiene el subíndice donde se encontró (búsqueda).
IN	contiene la opción a ejecutarse.
KB$	llave que se busca.
LI,LS	en salida : primer y último registro a mostrar. en busca : límite inferior y superior de la lista o sub-lista en que se busca.
MX	tamaño máximo de la tabla de directorio.
ND$(i)	arreglo con los nombres de las delegaciones (DIM(16)).
OK	salida búsqueda;0= no se encuentra;1= si se encuentra.
PR	si PR=1,salida a papel.
SN$	variable para captar respuesta SI o NO.
VI	si VI=1,salida a video.
X	variable para esperar un ENTER.
X1$,X2$	su contenido determina si la salida es de claves "P","C" o ambas.

NOTA : si VI=PR=0,indicará salida a etiquetas.

VARIABLES AUXILIARES

en CAPTURA DE DATOS :

B
DK$
DN$
DD$
DC$
DP
D1
D2

en BUSQUEDA :

NO

en ORDENAMIENTO :

O1	TK$
O2(i)	TN$
O3(i)	TD$
O4	TC$
O5	TP
I	T1
J	T2
MU$	

(E)

CLAVE : C01
JOSE RICARDO CIRIA MERCE
CIUDAD UNIVERSITARIA-IIMAS
COLONIA OXTOPULCO UNIVERSIDAD
4970 COYOACAN
$ 23,423
$432,421

CLAVE : C03
JUAN PEREZ-MARTINEZ
AV.ROSEDAL NO. 435 INTERIOR 203
COLONIA JARDINES DEL BOSQUE
52490
$367,900
$ 1,235

CLAVE : C47
JORGE ONTIVEROS JUNCO
AV.UNIVERSIDAD-2014-BRASIL-603
COLONIA INTEGRACION LATINOAMERICANA
4350 COYOACAN
$578,974
$ 34

CLAVE : C81
HECTOR ARRONA-URREA
FRANCISCO MORAZAN 701
COLONIA CENTRO
15810 VENUSTIANO CARRANZA
$999,999
$555,666

CLAVE : P01
JUAN ALEJANDRO JIMENEZ GARCIA
IXCATEOPAN NO. 62
COLONIA LETRAN VALLE
3650 BENITO JUAREZ
$ 0
$ 0

CLAVE : P43
CARLOS RAMOS LARIOS
DIRECCION GENERAL DE PROVEDURIA
COLONIA CIUDAD UNIVERSITARIA
4900 COYOACAN
$547,364
$976,532

CLAVE : P75
PROVEEDORA DE PAPELERIA ESPECIAL S.A.
VIENA NO. 133
COLONIA PROGRESO
5600 CUAJIMALPA
$ 971
$ 477

CLAVE : P77
HERIBERTO OLGUIN ROMO
RINCON DE LOS ABETOS 101
COLONIA RINCONADA COAPA
16780 XOCHIMILCO
$ 23
$ 43

CLAVE : C01 (12)
JOSE RICARDO CIRIA MERCE
CIUDAD UNIVERSITARIA-IIMAS
COLONIA OXTOPULCO UNIVERSIDAD
4970 COYOACAN

CLAVE : C03
JUAN PEREZ-MARTINEZ
AV.ROSEDAL NO. 435 INTERIOR 203
COLONIA-JARDINES-DEL-BOSQUE
52490

CLAVE : C47
JORGE ONTIVEROS JUNCO
AV.UNIVERSIDAD-2014-BRASIL-603
COLONIA INTEGRACION-LATINOAMERICANA
4350 COYOACAN

CLAVE : C81
HECTOR-ARRONA-URREA
FRANCISCO MORAZAN-701
COLONIA CENTRO
15810 VENUSTIANO CARRANZA

CLAVE : P01
JUAN ALEJANDRO JIMENEZ-GARCIA
IXCATEOPAN NO. 62
COLONIA LETRAN VALLE
3650 BENITO JUAREZ

CLAVE : P43
CARLOS RAMOS LARIOS
DIRECCION GENERAL DE PROVEDURIA
COLONIA CIUDAD UNIVERSITARIA
4900 COYOACAN

CLAVE : P75
PROVEEDORA DE PAPELERIA ESPECIAL S.A.
VIENA NO. 133
COLONIA PROGRESO
5600 CUAJIMALPA

CLAVE : P77
HERIBERTO OLGUIN ROMO
RINCON DE LOS ABETOS 101
COLONIA RINCONADA COAPA
16780 XOCHIMILCO

```
CLEAR 1000
10 REM   UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
                FACULTAD DE INGENIERIA
             DIVISION  DE  EDUCACION CONTINUA
                 SISTEMA * DIRECTORIO *
15 '          JOSE RICARDO CIRIA MERCE  1983
20 GOSUB 300:
   GOSUB 320:
   GOSUB 180
50 IF IN=1 THEN GOTO 160

60   GOSUB 320:
     GOSUB 350:
     GOSUB 370:
     ON IN GOTO 70,80,90,100,110,120,130,140
70   GOSUB 210:
     GOTO 150          '-1-SALIDA DEL SISTEMA
80   GOSUB 420:
     GOTO 150          '-2-ALTA
90   GOSUB 2120:
     GOTO 150          '-3-BAJA
100  GOSUB 1980:
     GOTO 150          '-4-CAMBIO
110  GOSUB 1300:
     GOTO 150          '-5-MUESTRA PANTALLA
120  GOSUB 1380:
     GOTO 150          '-6-LISTA
130  GOSUB 1400:
     GOTO 150          '-7-ETIQUETAS
140  GOSUB 2080:
     GOTO 150          '-8-BUSCA
150  GOTO 50:
     'END CASE
160  'END WHILE
170  GOSUB 320:
     PRINT "FIN DE LA EJECUCION":
     END

180  REM SUBRUTINA INICIO
190  INPUT "NUEVO DIRECTORIO (SI,NO)";SN$:
     IF SN$ <> "SI" THEN:
        GOSUB 1110
     ELSE
        C1=1:
     'END IF
200  RETURN

300  REM SUBRUTINA INICIALIZA
305  DIM ND$(16):
     GOSUB 940
310  EN$="D.E.C.F.I.      SISTEMA    DIRECTORIO" :
     E1$="FACULTAD DE INGENIERIA U N A M" :
     Z=0 :
     MX=10 :
     F1$="$$$$,$$$ ":
     RETURN

320  REM SUBRUTINA ENCABEZADO
330  CLS:
     PRINT STRING$(10,"*");E1$:
     PRINT STRING$(10," ");EN$:
     PRINT:
     RETURN
```

```
210 REM SUBRUTINA TERMINA
220 GOSUB 320:
    IF C1=1 THEN
       GOSUB 1150
    'NOELSE
    END IF
230 RETURN
```

```
340 REM SUBRUTINA MUESTRA MENU
350 PRINT"1-SALIDA DEL SISTEMA":
    PRINT"2-ALTA" :
    PRINT"3-BAJA":
    PRINT"4-CAMBIO" :
    PRINT"5-MUESTRA DIRECTORIO (PANTALLA)" :
    PRINT"6-LISTA DIRECTORIO" :
    PRINT"7-IMPRIME ETIQUETAS":
    PRINT"8-BUSCA"

360 RETURN
```

```
370 REM SUBRUTINA CAPTA OPCION
380 IN=0:
    PRINT@775,"TECLEE LA OPCION DESEADA !":
390 IF (IN>0) AND (IN<9) THEN 410
400 PRINT@802,"-------":
    PRINT@799," ":
    INPUT IN:
    GOTO 390
410 RETURN
```

```
940 REM SUBRUTINA CARGA DELEGACIONES
950 FOR I=1 TO 16:
       READ ND$(I) :
    NEXT I:
    RETURN
```

```
2030 DATA ALVARO OBREGON,AZCAPOTZALCO,BENITO JUAREZ
2040 DATA COYOACAN,CUAJIMALPA,CUAUHTEMOC,GUSTAVO A. MADERO
2050 DATA IZTACALCO,IZTAPALAPA,M. CONTRERAS,MIGUEL HIDALGO
2060 DATA MILPA ALTA,TLAHUAC,TLALPAN,VENUSTIANO CARRANZA
2070 DATA XOCHIMILCO
```

```
960 REM SUBRUTINA ESPERA ENTER
970 PRINT"PARA CONTINUAR OPRIMA < ENTER >";:
    INPUT X$
    RETURN
```

```
910 REM SUBRUTINA OBTIENE DELEGACION
920 IF (DE < 1) OR (DE > 16) THEN
        DL$=STRING$(10," ")
    ELSE
        DL$=ND$(DE)
    'ENDIF
930 RETURN
```

```
2160 REM SUBRUTINA PARAMETROS LISTA TODO
2170 X1$="C":
     X2$="P":
     RETURN
```

```
1110 REM SUBRUTINA CARGA DIRECTORIO
1120 PRINT"DE <-REWIND-> AL CASSETTE Y-":
     PRINT "PONGALO EN < PLAY > CON VOLUMEN =5":
     GOSUB 960:
1130 INPUT #-1,CD:
     FOR I=1 TO CD:
       INPUT #-1,MK$(I),MN$(I),MD$(I),MC$(I),MP(I),M1(I),M2(I):
     NEXT I
1140 RETURN
```

```
1150 REM SUBRUTINA  GUARDA DIRECTORIO
1160 PRINT"DE REWIND AL CASSETTE Y":
     PRINT"PONGALO EN RECORD":
     GOSUB 960:
     PRINT#-1,CD:
     FOR I=1 TO CD
1170   PRINT#-1,MK$(I),MN$(I),MD$(I),MC$(I),MP(I),M1(I),M2(I):
     NEXT I:
     PRINT"DIRECTORIO GRABADO...":
     RETURN
```

```
1980 REM SUBRUTINA CAMBIO
1990 GOSUB 320:
     PRINT ">>>>>CAMBIO<<<<<":
     GOSUB 520:
     KB$=DK$:
     GOSUB 1650
2000 IF OK <> 1 THEN
       PRINT"NO EXISTE LA CLAVE :";DK$
     ELSE
       LI=ID:
       LS=ID:
       VI=1:
       PR=0:
       GOSUB 860:
       GOSUB 1880:
     'END IF
2010 GOSUB 960:
     RETURN
```

```
1840 REM SUBRUTINA PREGUNTA CAMBIO
1850 INPUT"DESEA CAMBIARLO (SI,NO)-";SN$:
     CA=0
1860 IF SN$="SI" THEN
       CA=1
     'NOELSE
     ENDIF
1870 RETURN
```

```
1880 REM SUBRUTINA CAMBIA LLAVE
1890 PRINT"NOMBRE:",MN$(ID):
     GOSUB 1840
1895 IF CA THEN
       GOSUB 590:
       MN$(ID)=DN$:
       C1=1:
     'NOELSE
     END IF
1900 PRINT"CALLE Y NUM.:",MD$(ID):
     GOSUB 1840
1910 IF CA THEN
       GOSUB 650:
       MD$(ID)=DD$:
       C1=1:
     'NOELSE
     END IF
1920 PRINT"COLONIA:",MC$(ID):
     GOSUB 1840
1930 IF CA THEN
       GOSUB 720:
       MC$(ID)=DC$:
       C1=1:
     'NOELSE
     END IF
1940 PRINT"CODIGO POSTAL:",MP(ID):
     GOSUB 1840
1950 IF CA THEN
       GOSUB 790:
       MP(ID)=DP:
       C1=1:
     'NOELSE
     END IF
1960 PRINT"SALDO 1 : ",M1(ID):
     GOSUB 1840
1962 IF CA THEN
       GOSUB 820:
       M1(ID)=D1:
       C1=1:
     'NOELSE
     END IF
```

```
1964 PRINT"SALDO 2 : ",M2(ID):
     GOSUB 1840
1966 IF CA THEN
       GOSUB 840:
       M2(ID)=D2:
       C1=1:
     'NOELSE
     END IF
1970 RETURN
```

```
420 REM SUBRUTINA ALTA
430 GOSUB 920:
    PRINT ">>>>> ALTA <<<<<":
    GOSUB 530:
    KB$=DK$:
    GOSUB 1650
440 IF DK=1 THEN
        PRINT"YA EXISTE LA CLAVE :";KB$ :
    ELSE
        GOSUB 460 :
    'ENDIF
450 PRINT"CLAVE -";DK$;"- DADA DE ALTA":
    GOSUB 960:
    RETURN
```

```
460 REM SUBRUTINA VERIFICA ESPACIO DISPONIBLE
470 IF CD >= MX THEN
        PRINT"TABLA DEL DIRECTORIO LLENA" :
    ELSE
        GOSUB 490 :
    'ENDIF
480 RETURN
```

```
490 REM SUBRUTINA ADICIONA CLAVE
500 C1=1:
    CD=CD+1:
    MK$(CD)=DK$:
    GOSUB 590 :
    MN$(CD)=DN$:
    GOSUB 650:
    MD$(CD)=DD$:
    GOSUB 720 :
    MC$(CD)=DC$ :
    GOSUB 790 :
    MP(CD)=DP :
    GOSUB 820 :
    M1(CD)=D1 :
    GOSUB 840 :
    M2(CD)=D2 :
    GOSUB 1450 :

510 RETURN
```

```
1650 REM SUBRUTINA BUSCA LLAVE
1660 OK=0:
     NO=0
1670 IF CD=0 THEN
        OK=0
     ELSE
        GOSUB 1690
     'END IF
1680 RETURN

1690 REM SUBRUTINA ITERA BUSQUEDA
1700 LI=1:
     LS=CD
1710 IF (OK=1) OR (NO=1) THEN 1730
1720    GOSUB 1740:
        GOTO 1710:
     'END DO
1790 RETURN

1740 REM SUBRUTINA BUSCA
1750 IF (LI+1)=LS OR (LI=LS) THEN
        GOSUB 1770
     ELSE
        GOSUB 1800
     'END IF
1760 RETURN

1770 REM SUBRUTINA BUSCA EN DOS
1780 IF MK$(LI)=KB$
        ID=LI:
        OK=1:
        NO=0
     ELSE
        IF MK$(LI+1)=KB$
           ID=LI+1:
           OK=1:
           NO=0
        ELSE
           OK=0:
           NO=1:
        'END IF
     'ENDIF
1790 RETURN

1800 REM SUBRUTINA BUSCA EN MAS DE DOS
1810 I1=INT((LS+LI)/2)
1820 IF MK$(I1)=KB$
        ID=I1:
        OK=1:
        NO=0
     ELSE
        IF MK$(I1)>KB$
           LS=I1
        ELSE
           LI=I1
        'END IF
     END IF
1830 RETURN
```

```
520 REM SUBRUTINA CAPTA CLAVE
530 B=0
540 IF B=1 THEN GOTO 580
550 INPUT "CLAVE (C00=C99 , P00=P99 )  ";DK$
560 IF ((LEFT$(DK$,1)="C") OR
        (LEFT$(DK$,1)="P")) AND
        LEN(DK$)=3 THEN
        B=1
    'NOELSE
    ENDIF
570 GOTO 540
580 RETURN

590 REM SUBRUTINA CAPTA NOMBRE
600 B=0
610 IF B=1 THEN GOTO 642
620 INPUT "NOMBRE  ";DN$
630 IF LEN (DN$) > 0 THEN:
        B=1
    'NOELSE
    ENDIF
640 GOTO 610
642 RETURN

650 REM SUBRUTINA CAPTA CALLE Y NUMERO
660 B=0
670 IF B=1 THEN GOTO 710
680 INPUT "CALLE Y NUMERO  ";DD$
690 IF LEN(DD$) > 0 THEN
        B=1
    'NOELSE
    ENDIF
700 GOTO 670
710 RETURN

720 REM SUBRUTINA CAPTA COLONIA
730 B=0
740 IF B=1 THEN GOTO 780
750     INPUT "COLONIA  ";DC$
760     IF LEN (DC$) > 0 THEN
            B=1
        'NOELSE
        ENDIF
770 GOTO 740
780 RETURN

790 REM SUBRUTINA CAPTA CODIGO
800 INPUT "CODIGO POSTAL  ";DP
810 RETURN

820 REM SUBRUTINA CAPTA SALDO 1
830 INPUT "SALDO 1 =";D1:
    RETURN

840 REM SUBRUTINA CAPTA SALDO 2
850 INPUT "SALDO 2=";D2:
    RETURN
```

```
1450 REM SUBRUTINA ORDENA (' QUICK-SORT )
1460 01=1:
     02(01)=1:
     03(01)=CO
1470 IF 01<= 0 THEN GOTO 1610
1480     04=02(01):
         05=03(01):
         01=01-1
1490     IF 04 >= 05 THEN GOTO 1600
1500         I=04:
             J=05:
             MU$=MK$(INT((04+05)/2))
1510         IF I > J THEN GOTO 1580
1520             IF MK$(I)>= MU$ THEN GOTO 1540
1530                 I=I+1:
                     GOTO 1520:
                 'END DO
1540             IF MU$ >= MK$(J) THEN 1560
1550                 J=J-1:
                     GOTO 1540:
                 'END DO
1560             IF I<=J THEN
                     GOSUB 1020:
                     I=I+1:
                     J=J-1:
                 'NOELSE
                 ENDIF
1570             GOTO 1510:
             'END DO
1580         IF I < 05 THEN
                 01=01+1:
                 02(01)=I:
                 03(01)=05:
             'NOELSE
             ENDIF
1590         05=J:
             GOTO 1490:
         'END DO
1600     GOTO 1470:
     'END DO
1610 RETURN
```

```
1020 REM SUBRUTINA INTERCAMBIA I-J
1030 TK$=MK$(I) :MK$(I)=MK$(J) :MK$(J)=TK$
1040 TN$=MN$(I) :MN$(I)=MN$(J) :MN$(J)=TN$
1050 TD$=MD$(I) :MD$(I)=MD$(J) : MD$(J)=TD$
1060 TC$=MC$(I) :MC$(I)=MC$(J) :MC$(J)=TC$
1070 TP =MP (I) : MP (I)=MP (J) : MP (J)=TP
1080 T1 =M1 (I) :M1 (I)=M1 (J) :M1 (J)=T1
1090 T2 =M2 (I) :M2 (I)=M2 (J) :M2 (J)=T2
1100 RETURN
```

```
860 REM SUBRUTINA REPORTES
870 FOR I=LI TO LS
875 PRINT"ESTOY EN REPORTES"
880    IF (LEFT$(MK$(I),1)=X1$) OR
          (LEFT$(MK$(I),1)=X2$) THEN
          DE=INT(MP(I)/1000):
          GOSUB 910:
          GOSUB 1200:
          'NOELSE
          ENDIF
900 NEXT I:
    RETURN
```

```
1200 REM SUBRUTINA SALIDA
1210 IF VI=1
        GOSUB 1230:
     ELSE
        GOSUB 1260
     'END IF

1220 RETURN
```

```
1230 REM SUBRUTINA VIDEO
1235 CLS
1240 PRINT "CLAVE : ";MK$(I):
     PRINTMN$(I):
     PRINTMD$(I):
     PRINT "COLONIA : ";MC$(I):
     PRINTMP(I);" ";DL$:
     PRINT  USING F1$;M1(I):
     PRINT USING F1$;M2(I)
1245 GOSUB 960
1250 RETURN
```

```
1260 REM SUBRUTINA PAPEL Y ETIQUETAS
1270 LPRINT "CLAVE : ";MK$(I):
     LPRINT MN$(I):
     LPRINT MD$(I):
     LPRINT "COLONIA ";MC$(I):
     LPRINT MP(I);" ";DL$:

1280 IF PR=1 THEN
        LPRINT USING F1$;M1(I):
        LPRINT USING F1$;M2(I)
     'NOELSE
     END IF
1282 LPRINT:
     LPRINT
1285 RETURN
```

```
1400 REM SUBRUTINA MUESTRA ETIQUETAS
1410 VI=0:
     PR=0:
     PRINT"MONTAR ETIQUETAS":
     GOSUB 960:
     GOSUB 1320:
     RETURN
```

```
1380 REM SUBRUTINA MUESTRA PAPEL
1390 VI=0:
     PR=1:
     PRINT"MONTAR PAPEL DIRECTORIO":
     GOSUB 960:
     GOSUB 1320:
     RETURN
```

```
1300 REM SUBRUTINA MUESTRA VIDEO
1310 VI=1:
     PR=0:
     GOSUB 1320:
     RETURN
```

```
1320 REM SUBRUTINA PARAMETROS REPORTES
1330 LI=1:
     LS=CD:
     INPUT"C=>CLIENTES;P=>PROVEEDORES;A->AMBOS ";SN$:
     X1$="C":
     X2$="P"
1350 IF SN$="C" THEN
        X2$="X"
     ELSE
        IF SN$="P" THEN
           X1$="X" :
        'NOELSE
        ENDIF
     ENDIF
1360 GOSUB 860:
     RETURN
```

```
2120 REM SUBRUTINA BAJA
2130 GOSUB 2160:
     GOSUB 920:
     PRINT ">>>>> BAJA <<<<<":
     GOSUB 520:
     KB$=OK$:
     GOSUB 1650
2140 IF OK <> 1 THEN
          PRINT"NO EXISTE LA CLAVE :-";OK$:
          GOSUB 960
     ELSE
          LI=ID:
          L6=ID:
          VI=1:
          PR=0:
          GOSUB 860:
          GOSUB 980:
     'END IF
2150 RETURN
```

```
980 REM SUBRUTINA ELIMINA CLAVE
990 INPUT "ES LA CLAVE A DAR DE BAJA (SI,NO)  !";SN$
1000 IF SN$="SI" THEN
          C1=1:
          MK$(ID)="ZZZ":
          GOSUB 1450:
          CD=CD-1 :
     'NOELSE
     ENDIF
1010 RETURN
```

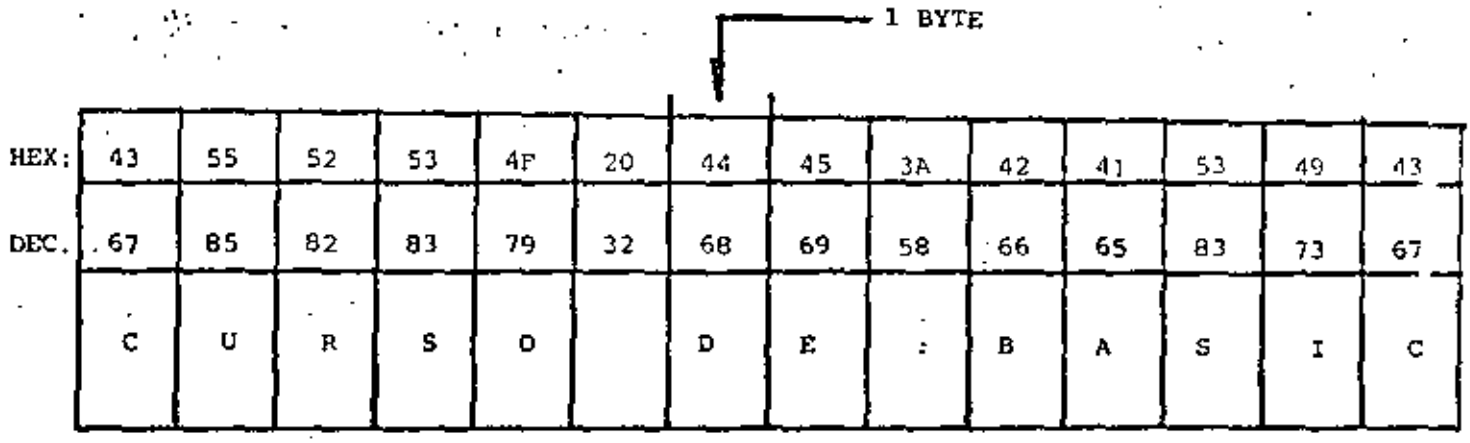
```
2080 REM SUBRUTINA BUSCA (MENU)
2090 GOSUB 2160:
     GOSUB 320:
     GOSUB 520:
     KB$=OK$:
     GOSUB 1650
2100 IF OK <> 1 THEN
          PRINT "NO EXISTE LA CLAVE :-";OK$:
          GOSUB 960
     ELSE
          LI=ID:
          L9=ID:
          VI=1:
          PR=0:
          GOSUB 860:
     'END IF
2110 RETURN
```

LENGUALE DE PROGRAMACION BASIC CON APLICACIONES

(PRIMERA PARTE)

ANEXOS

18 MARZO 1983



- LONGITUD MAXIMA : 255 CARACTERES
- ASIGNACION DE CONSTANTES "ENTRE COMILLAS"

  EJ. A$ = "BASIC"
- ASIGNACION VIA INPUT, INKEY$ y READ (DATA)

  SIN COMILLAS, EXCEPTO EN

  READ (DATA) QUE CONTENGA COMAS (,) o

  BLANCOS A LA IZQUIERDA
- SE CUENTA (al encender el quipo) CON 50 BYTES

  DE MEMORIA PARA STRINGS
- ESTO SE PUEDE MODIFICAR CON

  CLEAR n
- n = NUMERO DE BYTES REQUERIDOS

```
10 DIM TM(5):
   CL=16924:
   PRINT"DAME LOS VALORES DE :":
   PRINT"MES,DIA,A/O,HORA,MINUTO,SEGUNDO"
20 INPUT TM(0),TM(1),TM(2),TM(3),TM(4),TM(5)
30 FOR I=0 TO 5:
      POKE CL-I,TM(I):
  NEXT I
40 PRINT"VALORES DE TIEMPO CARGADOS : ";TIME$ :
   END
```

```
10 C=0:
   A$="X":
   PRINT"TECLEA UN CARACTER ALFA"
20 IF A$="A" THEN 40
30      C0=C0+1:
        PRINT"ACUMULADO : ";C0 :
        A$=INKEY$:
        GOTO 20:
   'END 00
40 PRINT"ACERTASTE ... ES UNA ";CHR$(34);"A";CHR$(34):
   END
```

DIRECTORIO DE ASISTENTES AL CURSO LENGUAJE DE PROGRAMACION BASIC CON APLICACIONES (DEL 18 DE FEBRERO AL 19 DE MARZO DE 1983)

NOMBRE Y DIRECCION	EMPRESA Y DIRECCION
1. ALEJANDRO ALMANZA TORRES Calle Trabajo Social No. 612 Edif. E-8 Depto. 103 México, D. F.	SECRETARIA DE AGRICULTURA Y RECURSOS HIDRAULICOS
2. LUIS IGNACIO L. ARCE LICONA Xola No. 125-401 Col. Alamos Deleg. Benito Juárez C.p. 03400 México, D. F. Tel: 6-96-47-48	PROCESAMIENTO GRAFICO ELECTRONICO, S. A. Av, Universidad No. 479 Col. del Valle Deleg. Benito Juárez C.P. 03100 México, D. F. Tel: 6-87-51-24
3. ENRIQUE ARENAS SANCHEZ Calz. Tlatilco No. 183 Col. Tlatilco Deleg. Azcapotzalco México, D. F. Tel: 5-41-44-69	U.N.A.M. Ciudad Universitaria México, D. F. Tel: 5-50-52-15 - 4611
4. RAFAEL CARRANZA VILLANUEVA Mochis No. 6 Zacatepec, Morelos Tel: 2-13-77	INSTITUTO TECNOLOGICO DE ZACATEPEC Zacatepec, Mor. Tel: 2-13-94
5. RAFAEL CASAL SILVA Niño Jesús 260 Edif. K-301 Unidad T. Tlalpan DELEG. Tlalpan C.P. 14620 México, D. F. Tel: 5-73-94-16	SEARLE
6. LUIS MIGUEL CONCHAS BOJALIL Cerro Macuiltepec 450 Col. Camp. Churubusco Deleg. Coyoacán C.P. 04200 México, D. F. Tel: 5-49-49-45	PROYECTOS Y ADMINISTRACIONES, S. A. Insurgentes Sur 1877-5o. Piso Col. Guadalupe Inn. México, D. F. Tel: 5-48-95-20

DIRECTORIO DE ASISTENTES AL CURSO DE PROGRAMACION BASIC CON APLICACIONES
(DEL 18 DE FEBRERO AL 19 DE MARZO DE 1983)

NOMBRE Y DIRECCION	EMPRESA Y DIRECCION
25. JOSE LUIS ROSALES OCHOA Libertad No. 4 Apaxco Edo. de México	ESTRUCTURAS CONSTRUCCIONES Y ACABADOS,S.A. Poniente 146 No. 916 Col. Industrial Vallejo México, D. F. Tel: 5-87-03-11
26. SOCORRO SANCHEZ GARCIA Cuauhtémoc 57 Col. Aragón Deleg. Gustavo A. Madero México, D. F. Tel: 5-77-83-80	CONSTRUCTORA HUARTECA, S. A. Petrarca 223-9o. Piso Col. Polanco México, D. F. Tel: 2-50-54-64
27. JOSE ANTONIO SANCHEZ RAMIREZ Pedro Loza 416 Guadalajara, Jal. Tel: 13-28-60	AUTOTRANSPORTES TRES ESTRELLAS DE ORO, S. A. de C. V. Av. 8 Esq. Poniente 114 s/n Magdalena de las Salinas C.P. 07760 México, D. F. Tel: 5-87-30-55