

LENGUAJE DE PROGRAMACION BASIC

TEMARIO

- 1.- INTRODUCCION AL PROCESO ELECTRONICO DE DATOS
- 2.- ELEMENTOS DE BASIC
- 3.- INTRODUCCION A LA MICRO COMPUTADORA TRS-80 MODELO III
- 4.- INTRODUCCION BASIC DE DECISION (IF)
- 5.- INTRODUCCION A LA PROGRAMACION ESTRUCTURADA
- 6.- INSTRUCCION BASIC DE REPETICION (FOR)
- 7.- FUNCIONES INTRINSECAS
- 8.- INSTRUCCIONES BASIC COMPLEMENTARIAS
- 9.- FORMATOS
- 10.- VARIABLES CON UN SUBINDICE
- 11.- VARIABLES CON MAS DE UN SUBINDICE
- 12.- MANEJO DE CARACTERES
- 13.- SUBPROGRAMAS
- 14.- PROGRAMACION ESTRUCTURADA EN BASIC
- 15.- APLICACIONES PRACTICAS
- 16.- MANEJO DE PANTALLA

LENGUAJE DE PROGRAMACION BASIC

MATERIAL ESCRITO

- 1.- EL UNIVERSO DE LAS COMPUTADORAS
- 2.- LA PROXIMA GENERACION DE COMPUTADORAS
- 3.- CONCEPTO DE UNA COMPUTADORA
- 4.- SISTEMAS DE NUMERACION
- 5.- GUIA DE REFERENCIA PARA LA MICROCOMPUTADORA TRS-80 MODELO III
- 6.- RESUMEN DE COMANDOS E INSTRUCCIONES TRS-8- MODELO III
- 7.- PROGRAMACION ESTRUCTURADA
- 8.- SISTEMA " DIRECTORIO"
- 9.- SISTEMA "MINIMOS CUADRADOS"
- 10.- EJEMPLOS DE PROGRAMAS EN BASIC

LENGUAJE DE PROGRAMACION BASIC

DISTRIBUCION DE TIEMPOS

HORARIOS:

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

LUGARES:

Viernes: Palacio de Minería Tacuba No. 5

Sábados: Centro de Cálculo Anexo (CECAF1) Anexo de la Facultad de Ingeniería
Ciudad Universitaria

PRIMER VIERNES

Tema: 1.- Introducción al Proceso Electrónico de Datos

Expositores: Carlos Augusto Ramos Larios
Heriberto Olguín Romo

PRIMER SABADO

Temas: 2.- Elementos BASIC

3.- Introducción a la Microcomputadora TRS-80 Modelo III

Expositores: Jorge Ontiveros Junco
José Ricardo Ciria Merce

SEGUNDO VIERNES

Temas: 4.- Introducción BASIC de decisiones (IF)

5.- Introducción a la Programación Estructurada

Expositores: Héctor Arrona Urrea
Juan Alejandro Jiménez García

SEGUNDO SABADO

Temas: 6.- Instrucción BASIC de repetición (FOR)

7.- Funciones Intrínsecas

Expositores: Héctor Arrona Urrea

Carlos Augusto Ramos Larios

TERCER VIERNES

Temas: 8.- Instrucciones BASIC complementarias

Expositores: Jorge Ontiveros Junco

Heriberto Olguín Romo

TERCER SABADO

Temas: 9.- Formatos

10.- Variables con un subíndice

Expositores: Héctor Arrona Urrea

Jorge Ontiveros Junco

CUARTO VIERNES

Temas: 11. Variables con más de un subíndice

12. Manejo de caracteres

Expositores: José Ricardo Ciria Merce

J. Alejandro Jiménez García

CUARTO SABADO

Temas: 13. Subprogramas

14. Programación Estructurada en BASIC

Expositores: Juan Alejandro Jiménez García

Carlos Augusto Ramos Larios

QUINTO VIERNES

Tema: 15. Aplicaciones Prácticas.

Expositores: José Ricardo Ciria Merce

Jorge Ontiveros Junco

QUINTO SABADO

Tema: 16. Manejo de Pantalla

Expositores: José Ricardo Ciria Merce

Heriberto Olguín Romo



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

LENGUAJE DE PROGRAMACION BASIC

EL UNIVERSO DE LAS COMPUTADORAS

SEPTIEMBRE, 1983

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 esta 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.

10001

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 múltiplos en columnas debajo de ellos. Al hacer girar los rodillos se podía multiplicar fácilmente y sin denotar 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 automatización, palabra derivada de automatización y acuñada por el norteamericano Delmer 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 automatización; 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 permitió 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 sí 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 están 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" ELECTRÓNICO

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 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 Analóg (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 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.

TAMAÑO Y VELOCIDAD

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 milionésimas 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.

00106

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, reemplaza 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.

MAQUINAS "INTELIGENTES"

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 consisten 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 insustituible 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 nada (ni nada) para hacerle al hombre.

ME DI OS DE 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". (4)

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.

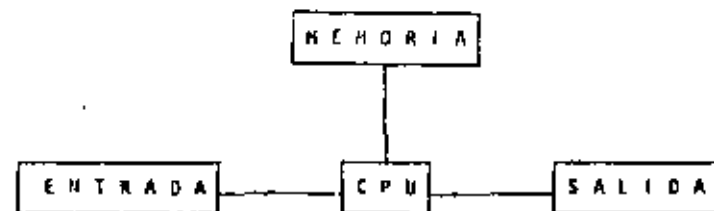
Hardware son los equipos electrónicos, mecánicos y electromecánicos que forman la estructura de la computadora. Esta parte se encarga de captar la información, de las operaciones aritméticas y lógicas, del almacenamiento de la información y de imprimir los resultados. Asimismo, está compuesta de:

- 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 desea.
- 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.



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

LENGUAJE DE PROGRAMACION BASIC

LA PROXIMA GENERACION DE COMPUTADORAS

SEPTIEMBRE, 1983

Enrique Calderón Alzati

(Comunicaciones de la Fundación Rosenblueth, Enero de 1982)

I. INTRODUCCION.

Aparecidas en los inicios de la década cincuenta, las computadoras constituyen hoy en día, uno de los instrumentos que mayor influencia ejerció 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 marcado 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.

11111

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 Integration), 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.

000014

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:	Instrumentos de cálculo.
Tecnología utilizada:	Tubos de vacío.
Unidades periféricas:	Memoria de cilindro magnético. 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 ³ ops/seg. Precio 100,000 a 2,5 millones US.
Modelos típicos:	IBM-650 Bendix-G15, Univac 5590, 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).
Alfabeto:	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 10^3 ops/seg. Precio 10^3 a 10^5 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 (DBMS).
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 10^4 ops/seg. Memoria secundaria 10^4 caracteres. Rango de precios 5×10^4 a 10^6 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 silicates).
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:	Irrestringido, mayúsculas y minúsculas, símbolos matemáticos, alfabeto Árabe, Japonés, etc.

Sistema operativo:	Proceso sin interrupción, comunicación entre máquinas, rutinas de recuperación, etc.
Facilidades adicionales:	Metaprosesadores, 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^{12} caracteres. Rango de precios 10^3 a 10^8 US.
Modelos típicos Grandes:	IBM-4330, Univac 1100, Burroughs 8-6900, 7900.
Medianos:	Prime 550 HP 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 legislación 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 cumple 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 secretarías 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 instrumentos 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 principiaron 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, quien 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, al desarrollo de LISP y otros lenguajes como Snobol y L⁶, hicieron factibles los trabajos en inteligencia artificial y manipulación de símbolos algebraicos, 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 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 instrumento de uso masivo, constituye hoy en día el objetivo a lograr para la siguiente generación de computadores.²

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.

²Figenbaum 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,

000021

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 microelectrónica, 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 ¿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.

000023



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

LENGUAJE DE PROGRAMACION BASICO

CONCEPTO DE UNA COMPUTADORA

SEPTIEMBRE, 1983

UNAM

*CONCEPTO
DE UNA
COMPUTADORA**

FACULTAD DE INGENIERIA

* TOMADO DE: "QUE HACEN LAS COMPUTADORAS Y COMO LO HACEN"
POR: I. B. M., Argentina.

507424

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.



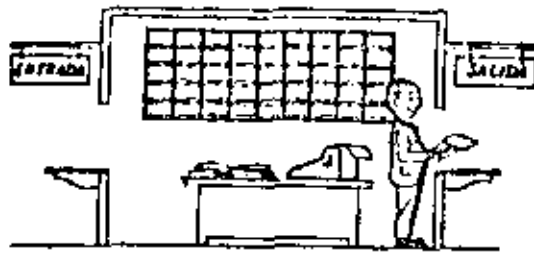
Este señor se llama Control. Trabaja en una pequeña habitación. Tiene a su disposición una máquina de calcular que suma, resta, multiplica y divide. Tiene también el señor Control un archivo parecido al casillero que existe en los trenes para clasificación postal. Hay, además, en la habitación, dos ventanillas 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.



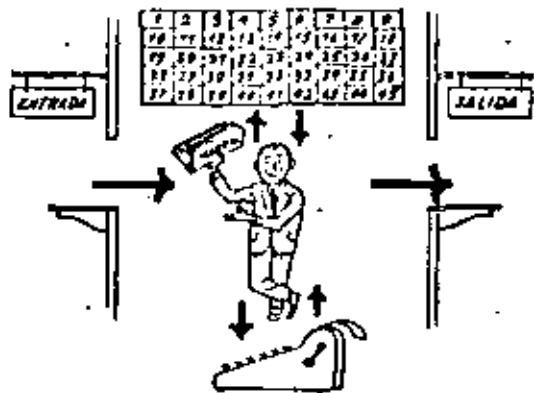
Una persona quiere saber el resultado de un complicado cálculo. Para ello, escribe ordenada, precisa y detalladamente, cada una de las operaciones que, en conjunto, integran ese cálculo, anota cada instrucción elemental en una hoja de papel y coloca todas las hojas en orden en la ventanilla "Entrada". El señor Control, al ver las hojas, lee en su manual que debe tomar esas hojas con instrucciones, una por una, y colocarlas en un archivo. Y así lo hace.



Una vez ubicadas todas las instrucciones en el archivo, el señor Control consulta 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 repeticiones 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 más concentrada (a razón de 80 a 2400 caracteres por pulgada de longitud) y de ser más 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 milisegundos.

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 Óptica de Caracteres Impresos: Puede leer un documento impreso por una máquina de escribir, o por una máquina de contabilidad o por la imprenta 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, números, ó signos especiales) cada uno.



Vemos aquí otra Unidad de Representación Visual, más evolucionada que la anterior, la comunicación hombre-máquina pueda 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 Óptica 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 X 30.5 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

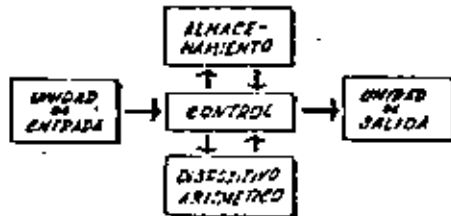
Registrador Analizador Fotográfico

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.

Finalmente, un dispositivo electrónico de control (representado por el señor control) ayudado de un programa especial o sistema operativo (representado por el manual del señor control), gobierna todas las operaciones de todas las unidades que componen la computadora.

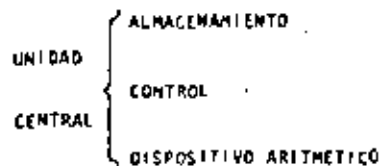
Habiendo descrito las partes que componen la computadora podemos mostrar el siguiente esquema que la representa:



O en forma más resumida :



Siendo :



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 (milionésimas de segundo).

El hecho de que las nuevas máquinas fueran miles de veces más rápidas que las anteriores, trajo aparejado 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 decal 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.

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 extra-ordinariamente.

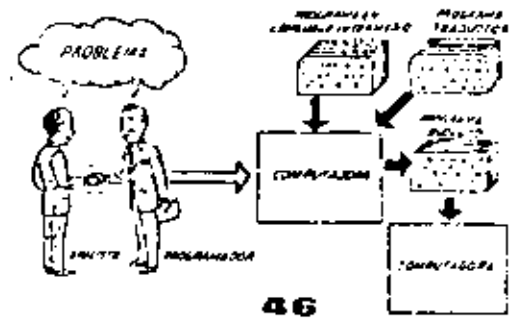
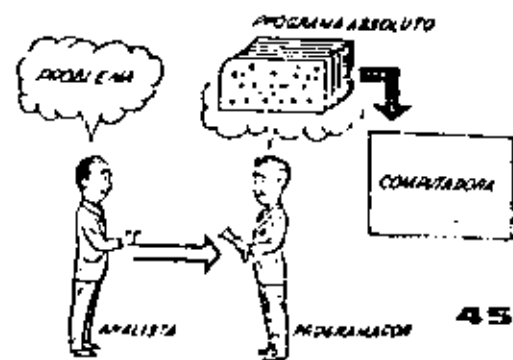
. Todas las operaciones del sistema se realizan en forma simultánea.

. 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. (Multi-proceso).

61



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". Los primeros computadores se "programaban" en este complejo lenguaje. Había entonces una enorme diferencia entre nuestro idioma y aquí según el cual 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.

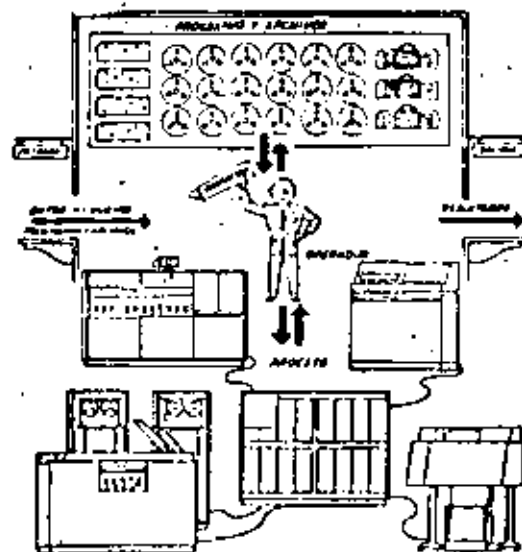
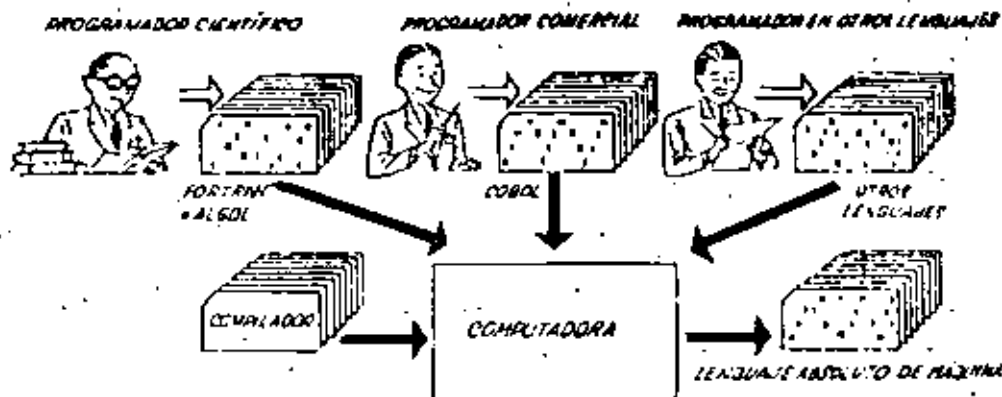
Se crearon, para solucionar el problema, lenguajes intermedios cada vez más parecidos a nuestro idioma. Es decir que cada nuevo lenguaje intermedio se acercaba más al problema y se alejaba más de la máquina. Para cada uno de estos lenguajes se creó un programa traductor llamado "Compaginador" o "Compilador", -- que tenía la misión de traducir el lenguaje intermedio al absoluto de máquina. Ahora, el analista y el programador "habían 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 constituyen 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 "entienda", 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

SISTEMA DE NUMERACION

SEPTIEMBRE, 1983

SISTEMAS DE NUMERACION

Ing. Heriberto Olguín Romo

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

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

donde:

r es la base del sistema de numeración.

$a_0, a_1, a_2, \dots, a_n$ 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:

$$\begin{aligned} N &= a_0 2^0 + a_1 2^1 + a_2 2^2 + a_3 2^3 + \dots + a_n 2^n \\ &= a_0 + a_1(2) + a_2(4) + a_3(8) + \dots + a_n(2^n) \end{aligned}$$

Los coeficientes $a_0, a_1, a_2, a_3, \dots, a_n$ 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 $a_0, a_1, a_2, a_3, \dots, a_n$; y arreglando estos coeficientes en el orden: $a_n, \dots, a_3, a_2, a_1, a_0$; 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:

0111110011001001

DIVISIONES SUCCESIVAS

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		
<u>- 128</u>	$128 = 2^7$	puede sustraerse
47		
<u>- 32</u>	$32 = 2^5$	" "
15		
<u>- 8</u>	$8 = 2^3$	" "
7		
<u>- 4</u>	$4 = 2^2$	" "
3		
<u>- 2</u>	$2 = 2^1$	" "
1		
<u>- 1</u>	$1 = 2^0$	" "
0		

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

$$175 = 10101111$$

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.

10101111

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

$$\begin{array}{r} 1100001 \quad 97 \\ + 1001110 \quad + 78 \\ \hline 10101111 \quad 175 \end{array}$$

Sustracción

-	0	1
0	0	1
1	0	0

+
y una deuda

$$\begin{array}{r} 10101111 \quad 175 \\ - 1001011 \quad - 75 \\ \hline 01100100 \quad 100 \end{array}$$

Multiplicación

x	0	1
0	0	0
1	0	1

$$\begin{array}{r} 11001 \quad . 25 \\ x \quad 111 \quad x . 7 \\ \hline 11001 \quad 175 \\ 11001 \\ \hline 11001 \\ \hline 1010111 \end{array}$$

División

÷	0	1
0	?	?
1	0	1

$$\begin{array}{r} 100011 \quad 35 \\ 101 \overline{) 10101111} \quad 5 \overline{) 175} \\ \underline{000} \quad \underline{25} \\ 0111 \quad 0 \\ \underline{0101} \\ 000 \end{array}$$

Cambio de un decimal fraccionario a binario:

$$\begin{array}{ll} 0.4375 \times 2 = 0.8750 & 0.0 \\ 0.8750 \times 2 = 1.7500 & 0.01 \\ 0.75 \times 2 = 1.5 & 0.011 \\ 0.5 \times 2 = 1.0 & 0.0111 \end{array} \quad 0.4375 = 0.0111_2$$

Cambio de un número binario fraccionario a decimal:

$$\begin{aligned} 0.0111 &= 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4} \\ &= \frac{1}{4} + \frac{1}{8} + \frac{1}{16} = \frac{4 + 2 + 1}{16} = \frac{7}{16} = 0.4375 \end{aligned}$$

b) Sistema Octal.

Un número escrito en base 8 toma la forma:

$$N = a_n 8^n + a_{n-1} 8^{n-1} + a_{n-2} 8^{n-2} + \dots + a_1 8^1 + a_0 8^0$$

$$= a_n + a_1(8) + a_2(16) + \dots + a_n(8^n)$$

Los coeficientes $a_0, a_1, a_2, \dots, a_n$ 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:

$$\begin{array}{r} 245 \\ 8 \overline{) 1967} \quad 7 \\ \underline{36} \\ 47 \\ \underline{7} \end{array}$$

$$\begin{array}{r} 30 \\ 8 \overline{) 245} \quad 5 \\ \underline{05} \end{array}$$

$$\begin{array}{r} 3 \\ 8 \overline{) 30} \quad 6 \\ \underline{6} \end{array}$$

$$\begin{array}{r} 0 \\ 8 \overline{) 3} \quad 3 \\ \underline{3} \end{array}$$

$$1967 = 3657_8$$

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 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
		<u>567</u>	<u>250</u>
		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:

- Sistema Binario.
- Sistema Octal.
- Sistema Decimal.
- 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 BASICO

TRS - 80 MODEL 111
MICROCOMPUTER SYSTEM

SEPTIEMBRE, 1983

TRS - 80 MODEL III
MICROCOMPUTER SYSTEM

GUIA DE REFERENCIA

CONTENIDO.

- I. ENCENDIDO
- II. PROPOSICIONES Y DECLARACIONES
- III. FUNCIONES
- IV. COMANDOS DE EDICION
- V. SUBROUTINAS ROM
- VI. DIRECCIONES IMPORTANTES DE RAM
- VII. CODIGOS DE CONTROL DE VIDEO
- VIII. MENSAJES DE ERROR
- IX. CARACTERES ESPECIALES
- X. OPERADORES

TRS - 80 MODEL III
MICROCOMPUTER SYSTEM

GUIA DE REFERENCIA

1. 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

Mos da una secuencia automática de líneas numeradas.

AUTO Inicia en 10 y se incrementa en 10

AUTO,5 Inicia en 5 y se incrementa en 5

AUTO,5 Inicia en 0 y se incrementa en 5

AUTO,50, Inicia en 50 y se incrementa en 10

CLEAR n Pag. 126

Inicializa todas las variables a cero (únicamente CLEAR).
Con argumento reserve 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.

CLOAD "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 encuentra.

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 en cuenta el primer carácter encontrado.

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

DATA Pag. 142.

Almacena datos que serán accedidos 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 DEFDDL 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 ALFAF 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 H,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.

Quando 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 o 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"DAHE TU NOMBRE";NOS

INPUT #-! 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 = 0+C
```

LIST secuencia. Pag. 11, 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.

LIST 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 brincaré 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, sólo 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 coma 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 dieciséis 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) "NOLA"
```

```
PRINT TAB(2*Y) "NOLA;TAB(2*X+5)NS"
```

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 primera 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 carácter. La longitud del campo será igual al número de espacios entre los signos % más dos.

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

Cualquier otro carácter que se utilice, simplemente se imprimirá como tal en la posición que ocupa.

Ejemplos:

```
10 AS="###.P"
```

```
20 PRINT USING AS;X
```

```
PRINT USING "##$.P";COSTO
```

```
PRINT USING "##.[ [ [ ";MON
```

```
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 deseas 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,NS
```

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 327. 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.

111. 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 Operation 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) = 12 PRINT "12"
```

- ASC (str) Pag. 164.

Regresa el código ASCII del primer carácter del string especificado.

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

```
PRINT ASC("A")
```

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

- ATN (x) Pag. 179.

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

```
100 Y = ATN (3/2)
```

- 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 carácter 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 fue 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 carácter teclado mediante un chequeo del tablero. Cuando no se tecldea 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 string especificado de longitud de la longitud determinada y a partir de la posición dada.

```
PRINT MID$(M$,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)
```

• RIGHTS (str,c)

Pag. 168.

Regresa los últimos c caracteres del string especificado.

```
PRINT RIGHTS("A",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 TO 20:PRINT RND(1):NEXT
```

• SIGN (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(H*B - 4*A*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")
```

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

• TAN (x) Pag. 183.

Cálcula la tangente del argumento dado en radianes.

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

• TIMES Pag. 170.

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

```
PRINT 600, TIMES
```

• USR (x) Pag. 191.

Permite llamar una subrutina en lenguaje de máquina, y después continuar la ejecución del programa BASIC. El argumento puede ser nudo o se puede utilizar para comunicación entre la subrutina y el programa BASIC. Ver páginas 59 a 60 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 tecléa 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 teclados, sin esperar que se presione la tecla ENTER.

Para mayores detalles, consultar las páginas 195 a 201, del Manual de Referencia (TAS-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.

- nK 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. SUBROUTINAS ROM.

La ROM (Memoria solo lectura) de la modelo III, contiene muchas subrutinas que pueden ser llamadas por un programa ensamblador 280 o por un programa BASIC, por medio de la funciónUSR.

Mayor información se encuentra en las páginas 60 a 80 y 191 del manual "TRS-80 Model III, Operation 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 42339/X'3033'

Muestra la fecha con que fue 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'1719'

Estando en un programa en lenguaje de máquina, 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'01FB'

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'002L'

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'004D'

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

\$KBBK 653/x'028p'

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

• SALIDA A IMPRESORA.

\$PRCHAR 59/x'0030'

Transfiere un carácter a la Impresora, si esta no está 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.

\$VCHAR 51/x'0033'

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

\$VCLS 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 carácter de la interfase RS232C.

\$RSTX 85/x'0055'

Transmite un carácter 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 II). 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 carácter ASCII que representa el 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 USA, 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 TMS-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	UC	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 fue 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 registrarse antes un ON ERROR GOTO.
20	UE	Se intentó 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 (cassetera).

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 precedencia:

+ & [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

RESUMEN DE COMANDOS E INSTRUCCIONES TRS-80 MODELO III

SEPTIEMBRE, 1983

Q U I C K

R A D I O S H A C K

SISTEMA TRS-80 MODELO III

MICROCOMPUTADORA RADIO SHACK

SISTEMA TRS-80 MODELO III

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 cassette (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.

115

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 carácter 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 precisión.
DEFDBL V,X-Z
- DEFINT
Define variables de tipo entero.
DEFINT A,I-M
- DEFSNG
Define variables de precisión simple.
DEFSNG I,W-Z

DEFSTR
Define variables de tipo string.
DEFSTR C:L-2

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≤n≤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 letreos(s) y/o variable(s).
LPRINT CAPS."ES EL CAPITAL DE";ST\$

LPRINT TAB
Mueve el carro de la impresora a la posición especificada.
LPRINT TAB(25)"REPORT"

LPRINT USING
Imprime con el formato especificado, números 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≤p≤255).
OUT 255, 0

00000000

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 letras 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 "**####"; 64.0

\$\$ Signo flotante de pesos.
PRINT USING "\$\$###.##"; 118.6735

**\$ Signo flotante de pesos; llena los primeros espacios con asteriscos.
PRINT USING "**+\$#.##"; 8.333

[Formato para exponentes. Presione la tecla t para que aparezca este carácter.
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 "###.##-"; -8126.42

! Imprime el primer carácter 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 SS READ MM\$,EDAD

REM
Comentario; instrucción que indica a la computadora que debe ignorar el resto de la línea. El apóstrofo (') 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 AT, entonces teclee el nombre del archivo o /dirección.

SYSTEM

TROFF

Anula la función TRACE.

TROFF

TRON

Prende la función TRACE.

TRON

CODIGO DE CONTROL DE VIDEO

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.

OPERADORES

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), sustracción.
- < , > , = , <= , >= , <> 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 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.

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 SPACEBAR Mueve el cursor n espacios a la derecha.
- n ← Mueve el cursor n espacios a la izquierda.

TECLAS DE CONTROL

- ← Borra el último carácter tecleado, regresa el cursor un espacio.
- SHIFT ← Borra la línea.
- BREAK Interrumpe alguna actividad y regresa al nivel de comandos.
- CTRL 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 (?) Estafifica la ejecución de un programa (pausa). Para continuar presione alguna tecla.

201142

FUNCIONES

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(15)

- 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(x#3)

- CHR\$(c)

Proporciona el caracter equivalente en código ASCII, control o código de gráficos.

PS=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 precisión simple.

FC = CSNG(TM#)

- 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 todas las 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(c5)

- INKEY\$

Proporciona al caracter teclado, si está disponible.

AS = 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.

PS = LEFT\$(MS,7)

- LEN(str)

Proporciona el número de caracteres del string.

X = LEN(SENS)

- LOG(x)

Calcula el logaritmo natural de x.

Y = LOG(x)

- MEM

Proporciona la cantidad de memoria disponible.

PRINT MEM

000043

- 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(16520)
- 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)
- RIGHTS(str, c)
Proporciona la parte derecha del string a partir de c.
ZIPS = RIGHTS(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.
SS = 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,"7") B\$ = STRING\$(125,63)
- TAN(x)
Calcula la tangente cuyo argumento es en radianes.
Y = TAN(x)

- TIMES
Proporciona el tiempo (con formato de 24 horas) y el dato es un string de 17 caracteres.
AS = TIMES
- 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))

MENSAJES DE ERROR

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	Línea indefinida.
9	BS	Subíndice fuera de rango.
10	DD	Arreglo dimensionado nuevamente.
11	ZD	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	NA	No hay una instrucción RESUME.
19	RW	Instrucción RESUME SIN ERADA.
20	UE	Error indefinido.
21	HO	Error de operandos.
22	FD	Archivo de datos mal salvado.
23	LJ	Solo para el sistema de Basic con disco.

DIRECCIONMIENTOS 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 nº0: Letras mayúsculas y minúsculas nº1: Sólo letras mayúsculas	Mayúsculas
16412	401C	Parpadeo del cursor nº0: Parpadea nº1: No parpadea	0
16416	4020	Localidad del cursor Dos bytes: LSB, MSB	N/A
16419	4023	Carácter del cursor Código ASCII (0-255)	176
16424	4028	Máximo de líneas/página más uno	67
16425	4029	Máximo 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 nº0: No espera nº1: Espera	Espera
16913	4211	Switch de la velocidad de grabación en cassette nº0: 500 bauds nº1: 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 \$ROUTC Dos bytes de E/S designados.	N/A
16930	4222	Dispositivo de default \$ROUTC Dos bytes de E/S designados.	N/A

SUBROUTINAS DE MEMORIA ROM Z-80

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 funciónUSR. 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	\$KBCCHAR	Checa el carácter presionado.
51	0033	\$VDCHAR	Aparece un carácter.
59	003B	\$PRCHAR	Imprime un carácter.
64	0040	\$KBLINE	Espera un comando.
73	0049	\$KBWAIT	Espera un carácter.
80	0050	\$RSRCV	Recibe un carácter desde RS-232-C.
85	0055	\$RSTX	Transmite un carácter 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	\$PRSCRN	Imprime el contenido de la pantalla.
539	0210	\$VDLINE	Aparece una línea.
545	0235	\$CSIN	Lectura de un byte en cassette.
612	0264	\$CSOUT	Salida de un byte en cassette.

LOCALIDAD		CONTENIDO	FUNCION
DEC	HEX		
647	0287	\$CSHWR	Escribe la cabeza del cassette.
653	0280	\$KBRK	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.



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

LENGUAJE DE PROGRAMACION BASICO

PROGRAMACION ESTRUCTURADA

SEPTIEMBRE, 1983

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 OSCURECE 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 CARACTER.
- 5) PARA SAANGRAR 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.

160

CONVENCIONES DE CODIFICACION

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, DEFSTR, DIM, END, ERROR, GOSUB,
 INPUT, LET, ON/OFF, OUT, POKE, PRINT, RANDOM, READ, REM, RESTORE, RESET,
 RETURN, SET, STOP.

B) DO WHILE

```
30  
IF A > B THEN 70 :
```

```
35 C = D :
```

```
PRINT :
```

```
·  
·  
·
```

SE PERMITEN VARIAS LINEAS FISICAS
EN EL RANGO DEL DO WHILE

```
GO TO 30
```

```
ENDDO. FIN DEL DO
```

```
70
```

```
F = G
```

```
·  
·  
·
```

C) DO UNTIL

```
80
```

```
FOR I = 1 TO N STEP 2 :
```

```
J = I+1 :
```

```
K = I*J :
```

```
·  
·  
·
```

SE PERMITEN VARIAS LINEAS FISICAS
EN EL RANGO DEL DO UNTIL

```
NEXT I
```

D) IF THEN ELSE

20

IF A < B THEN

C = D+E:

F = G-H

ELSE

PRINT I,J

ENDIF

FIN DEL IF

LA INSTRUCCION IF THEN ELSE DEBE
CABER TOTALMENTE EN UNA LINEA
(? SIN ERROR).

E) CASE

ON I GO TO 20, 30, 40

20 A = B * C:

D = E:

:

GO TO 50

30 G = H:

:

:

GO TO 50

40 I = J + K:

:

GO TO 50

50

L = M + N

:

:

:

:

SE PERMITEN VARIAS LINEAS FISICAS
EN CADA RANGO

PUEDA OMITIRSE

MANEJAR CON EXTREMA CUIDADA

- GO TO
- AN ERROR GO TO Y
- RESUME

LAS CUALES SON FIGURAS DE TRANSFERENCIA
INCONDICIONAL Y CAUSAS DE MUCHOS PROBLEMAS.

LA FIGURA GO TO NUNCA ES NECESARIA.

LAS INSTRUCCIONES AN ERROR GO TO Y RESUME PUEDEN SER
DE MUCHA AYUDA SI SE UTILIZAN PARA MANEJAR LAS CONDICIONES
DE ERROR EXCLUSIVAMENTE.

COMBINACION DE ESTRUCTURAS BASICAS

A) DO DENTRO DE IF

10

```
IF A <> B THEN
  FOR J=1 TO 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 DENTRO DE DO

30

FOR K= 1 TO 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 FOR PUEDE TENER VARIAS
LINEAS FISICAS

NO PUEDE LLEVAR 'ENDIF' SI EL NEXT
ESTA EN LA MISMA LINEA FISICA
(AL COLLOCAR RLM 6 ' , EL RESTO DE LA
LINEA FISICA SE CONSIDERA COMENTARIO)

C) IF DENTRO DE IF

20

IF A THEN

IF B THEN

D= 1:

C= D*2

ELSE

C=E

ELSE

C=F:

R=T:

H=Z

'ENDIF

TODO EL TEXTO DEBE CABER EN UNA
LINEA FISICA (IF THEN ELSE).

EN EL CASO QUE EL IF INTERNO NO
TENGA ELSE, DEBERA INCLUIRSE EL ELSE
INTERNO CON UNA INSTRUCCION NUDA COMO A=A
PARA FORZAR QUE EL ELSE DE LA

COLUMNA UNO CORRESPONDA AL IF
EXTERNO.

EL IF INTERNO NO PUEDE LLEVAR
'ENDIF (VER CASO ANTERIOR)


```

10 *DESGLOSE DE CANTIDADES TIPOCRA VERSION CAL FEB. 68
20 GOSUB 120 *VALORES INICIALES A LAS VARIABLES
30 INPUT CP
40
50 IF CP <= 0 THEN 100
55 LPRINT "LA CANTIDAD", CP, " SE DESGLOSA EN :
  N1=C1N2=C1N3=C1N4=C1N5=C1N6=C1N7=C1
  T=T+CP
60 *DESGLOSE EN 1000(GOSUB130),500(160),100(180),50(210)
  20(230),10(250) Y 5(270)
70 GOSUB 130:
  GOSUB 160:
  GOSUB 180:
  GOSUB 210:
  GOSUB 230:
  GOSUB 250:
  GOSUB 270:
80 GOSUB 290 *ACUMULA PARCIALES EN TOTALES
90 GOSUB 300:
  INPUT CPI
  GOTO 40
100 GOSUB 300 *IMPRIE TOTALES FINALES
110 END

120
V1=1000:V2=500:V3=100:V4=50:V5=20:V6=10:V7=5:V8=5:
Q=0:U=1:D=2:
RETURN

130
IF CP<=V1 THEN 150
140 CP=CP-V1:
  N1=N1+Q:
  GOTO 130:
*FIN DE
150 RETURN

```

D) USO DE GOSUB EN IF QUE REQUERIRIAN MAS DE UNA LINEA FISICA

```

10
IF A THEN
  GOSUB200
ELSE
  GOSUB300
ENDIF
20
.
.
.
200
IF B THEN
  D=1:
  C=D*2
ELSE
  C=E:
ENDIF
210 RETURN
300

```

SIEMPRE CABE EN UNA SOLA LINEA FISICA

ESTE IF DEBE CABER EN UNA SOLA LINEA, SI ESTO NO FUERA TENDRIA POSIBLE, SE REPETIRIA EL USO DEL GOSUB (QUIZA GOSUB 400 Y GOSUB 500) A ESTE NIVEL Y EN LOS QUE LE SIGAN DE SER NECESARIO

DEBE SER OTRA LINEA FISICA (EN IF QUE EJECUTA UN THEN AL TERMINAR SALTA A LA SIGUIENTE LINEA)

C= F:
R= T:
M= Z:
RETURN

PUEDA SER LA MISMA LINEA FISICA

```

150
IF CP > V2 THEN
  CP=CP-V2:
  N2=U
ENDIF
170 RETURN

180
IF CP <= V3 THEN 200
190 CP=CP-V3:
  N3=N3+U:
  GOTO 180:
ENDIF
200 RETURN

210
IF CP > V4 THEN
  CP=CP-V4:
  N4=U
ENDIF
220 RETURN

230
IF CP > V8 THEN
  CP=CP-V8:
  N5=D
ELSE
  IF CP > V5 THEN
    CP=CP-V5:
    N5=U
  ENDIF
ENDIF
240 RETURN

250
IF CP > V6 THEN
  CP=CP-V6:
  N6=U
ENDIF
260 RETURN

```

```

270
IF CP > V7 THEN
  CP=CP-V7:
  N7=U
ENDIF
280 RETURN

290 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

300 IF >N1 0 THEN LPRINTN1: "DE $1000"
310 IF >N2 0 THEN LPRINTN2: "DE $500"
320 IF >N3 0 THEN LPRINTN3: "DE $100"
330 IF >N4 0 THEN LPRINTN4: "DE $50"
340 IF >N5 0 THEN LPRINTN5: "DE $20"
350 IF >N6 0 THEN LPRINTN6: "DE $10"
360 IF >N7 0 THEN LPRINTN7: "DE $5"
370 IF >CP 0 THEN LPRINTCP: "DE $1"
380 RETURN

390
N1=M1: N2=M2: N3=M3: N4=M4: N5=M5: N6=M6: N7=M7: CP=M8:
LPRINT "TOTALES":
GOSUB 100:
LPRINT "SUMA TOTAL", T
RETURN

```

LA CANTIDAD 1978 SE DESGLOSA EN :

1 DE \$500
4 DE \$100
1 DE \$50
2 DE \$20
1 DE \$5
5 DE \$1

LA CANTIDAD 1574 SE DESGLOSA EN :

1 DE \$1000
1 DE \$500
1 DE \$50
1 DE \$20
4 DE \$1

LA CANTIDAD 567 SE DESGLOSA EN :

1 DE \$500
1 DE \$50
1 DE \$10
1 DE \$5
2 DE \$1

TOTALES

1 DE \$1000
3 DE \$500
4 DE \$100
3 DE \$50
3 DE \$20
1 DE \$10
2 DE \$5
11 DE \$1
SUMA TOTAL

\$3141

```
10 CLS
20 'ESTE PROGRAMA ORDENA UN VECTOR DE MENOR A MAYOR
30 DIM A(20)
40 INPUT "TAMANO 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
50
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
  60 NEXT J:
NEXT I:
LPRINT "EL VECTOR ORDENADO ES:":
FOR I = 1 TO TV:
  LPRINT A(I):
NEXT I:
70 GOTO 40
```

```
EL VECTOR ORIGINAL ES:
657
12.56
8975
25
4
EL VECTOR ORDENADO ES
4
12.56
25
657
8975
```

EJEMPLO DE DOCUMENTACION

PROGRAMA: Desplaz de cantidades

PROPÓSITO: Desplaza de un número cualquiera de cantidades en determinadas

de 1000,500,100,50,20,10,5 y 1

AUTOR: Carlos A. Ramos L.

FECHA: Enero, 1980

MOTIVADO EN: 1) Ganeta / 3 Q22-Q30 -4*

2) Dices / 7 "DESCLO/BAS"

DATOS NECESARIOS: Cantidades a desplazarse

RESULTADOS: Cada cantidad leída se imprime junto con su desplaz. Al final se

escriben la suma de las cantidades leídas junto con el desplazamiento

MÉTODO: Cada cantidad se va decrementando o moviendo que se van encan-

tando cantidades en las que se pueda descomponer. Una cantidad-

también se utiliza solo cuando la cantidad restante o descomponer la

encade.

EJEMPLO 50=20+20+5+1+1+1+1+1

```

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 "TAMANO DEL VECTOR A",TA:
FOR I=1 TO TA:
  INPUT A(I):
NEXT I:
50 INPUT "TAMANO DEL VECTOR B",TB:
FOR I=1 TO TB:
  INPUT B(I):
NEXT I:
60 CLS:PRINT "EL VECTOR A ES:"
FOR I=1 TO TA:
  PRINT A(I):
NEXT I:
70 PRINT "EL VECTOR B ES:"
FOR I=1 TO TB:
  PRINT B(I):
NEXT I:PRINT "LA CLASIFICACION DE LOS ELEMENTOS ES:"
A(TA+1)=100000:B(TB+1)=100000:
I=1:J=1:
80 IF A(I)=100000 AND B(J)=100000 THEN 140
90 IF A(I)<=B(J) THEN 99
99 GOSUB 150:
  GOTO 70:
  'ENDDO
100 IF A(I)>=B(J) THEN 110
100 GOSUB 160:
  GOTO 90:
  'ENDDO
110 IF A(I) > B(J) OR A(I)=100000 THEN 130
120 GOSUB 170:
  GOTO 110:
  'ENDDO
130 GOTO 60
  'ENDDO
140 GOTO 40
150 PRINT B(J),"SOLO EN B":J=J+1:RETURN
160 PRINT A(I),"SOLO EN A":I=I+1:RETURN
170 PRINT A(I),"EN AMBOS":I=I+1:J=J+1:RETURN

```

EL VECTOR A ES:

2 3 4 5

EL VECTOR B ES:

5 6 7 8

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

```

OBSERVACIONES: Una copia del listado puede encontrarse en la carpeta 6 de la biblioteca.

BIBLIOGRAFIA:

(Ver hoja anterior)

Entrada
 ? 1000
 ? 1574
 ? 567
 ? 0
 Salida

EJEMPLO:

debe ser cero, lo cual indica el fin de las cantidades.
 Entrada de datos: Los datos se proporcionan por la pantalla. La última cantidad

Lenguaje: BASIC RADIO SHACK LEVEL II

USO DEL PROGRAMA:

RESTRICCIONES: Las cantidades deben ser múltiplos de un peso (sin centavos)

F U N C I O N A M I E N T O I N T E R N O

Nombre de la variable	Significado
C	Contorno de Valor
D	0
V1	1
V2	1000
V3	500
V4	100
V5	50
V6	20
V7	10
V8	5
CF	40
I	Cantidad a devolver
N1	Suma de cantidades
N2	Contador por cantidad de billetes de
N3	500
N4	100
N5	50
N6	20
N7	10
N8	5
M1	Contador total de billetes de
M2	1000
M3	500
M4	400
M5	200
M6	100
M7	50
M8	20
M9	10
M10	5
M11	monedas
M12	Contador total de billetes de
M13	1000
M14	500
M15	100
M16	50
M17	20
M18	10
M19	5
M20	monedas
M21	Contador por cantidad de billetes de
M22	500
M23	100
M24	50
M25	20
M26	10
M27	5
M28	monedas
M29	Contador total de billetes de
M30	1000
M31	500
M32	400
M33	200
M34	100
M35	50
M36	20
M37	10
M38	5
M39	monedas
M40	Contador total de billetes de
M41	1000
M42	500
M43	400
M44	200
M45	100
M46	50
M47	20
M48	10
M49	5
M50	monedas

```

IF p THEN
  A function
  B function
  IF q THEN
    INCLUDE t-test
  ELSE
    C function
    DOWHILE T
    D function
    ENDDO
    CALL
  ENDF
  IF v THEN
    J function
  ELSE
    ENDF
  ENDIF
  IF x THEN
    H function
  ELSE
    L function
  ENDF
  ENDF
  K function
ENDIF

```

t-test

```

-IF t THEN
  G function
  DOWHILE u
  H function
  ENDDO
  I function
  ELSE
  ENDF

```

SEGMENTACION DE PROGRAMAS

U.N.A.M. 1969 HOJA DE DISEÑO-DOCUMENTACION H I P O C E C A F I I

SISTEMA: _____ AUTOM: _____
 SUBSISTEMA: _____ REFERENCIA: _____ HOJA DE
 APLICACION: _____ FUNCION: _____ FECHA: _____

ENTRADA

Cualquier problema de programación
 Programación estructurada
 Parado código

PROCESO

Hojas para a descomposición de un programa:
 - Muestra la hoja de pseudo código por separado
 - Contiene estructuras completas.
 - Si se requiere probar condicio- nes, el separado, contendrá sola- mente:
 - FUNCION y/o
 - DOWHILE y/o
 - ENDDO y/o
 - CALL y/o
 - Si se requiere probar condicio- nes, el separado, contendrá sola- mente, figuras de FLOWCHART.

SALIDA

Un programa propio formado por una jerarquía de segmen- tos. A su vez, cada segmento es un programa propio y su lógica es narrada con pseudo código.

CAPACTERISTICAS 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 INTERIAS.
- 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 INT" EN LOGICAS INTERIAS POR DESARROLLAR.

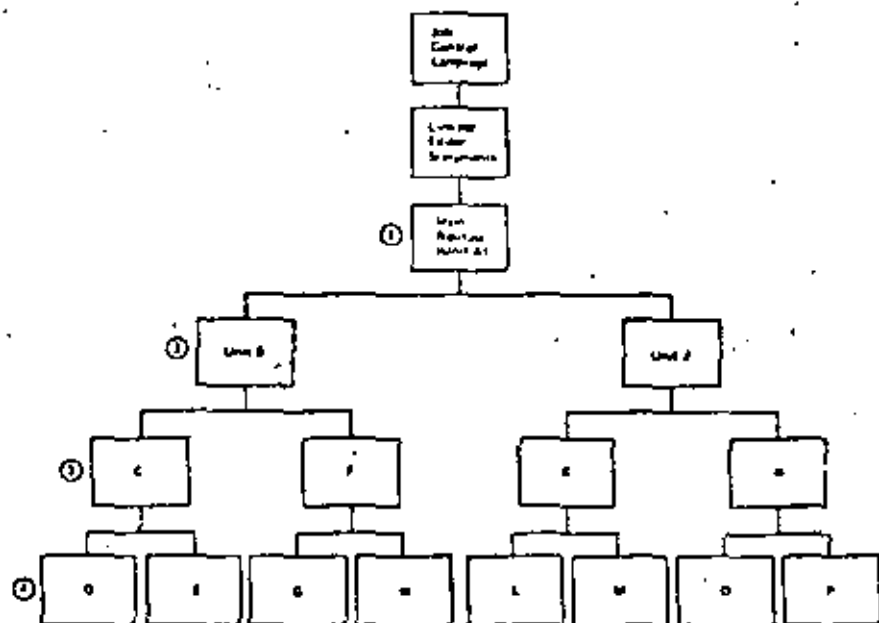


Figure 3. An example of a program directory

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 TEMA DE LA ESTRUCTURA GARANTIZA LA EXISTENCIA DEL MÓDULO POR DESARROLLAR.
- EL PROGRAMA ADQUIERE LA ESTRUCTURA DE UN ÁRBOL BIEN DEFINIDO.
- LAS RAMAS DEL ÁRBOL PUEDEN DESARROLLARSE EN FORMA TOTALMENTE INDEPENDIENTE (FACILITA EL TRABAJO EN EQUIPO).
- ES FÁCIL LOCALIZAR RUTINAS QUE SE REPITEN.

SISTEMA:		AUTOR:	HOJA DE
SUBSISTEMA:		REFERENCIA:	FECHA:
APLICACION:		FUNCION:	
ENTRADA	PROCESO	SALIDA	
		PARA PRODUCIR LOS RESULTADOS DESIADOS YA SEAN FINALES O INTER- MEDIOS	
PARA LOGRAR QUE SEA NECESARIO	PARA HACERLO QUE SE REQUIERA		

HIPO (HIERARCIY 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 O SISTEMA.
- PERMITE TENER UNA VISION GLOBAL O BIEN CONSULTAR EL MEJOR 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

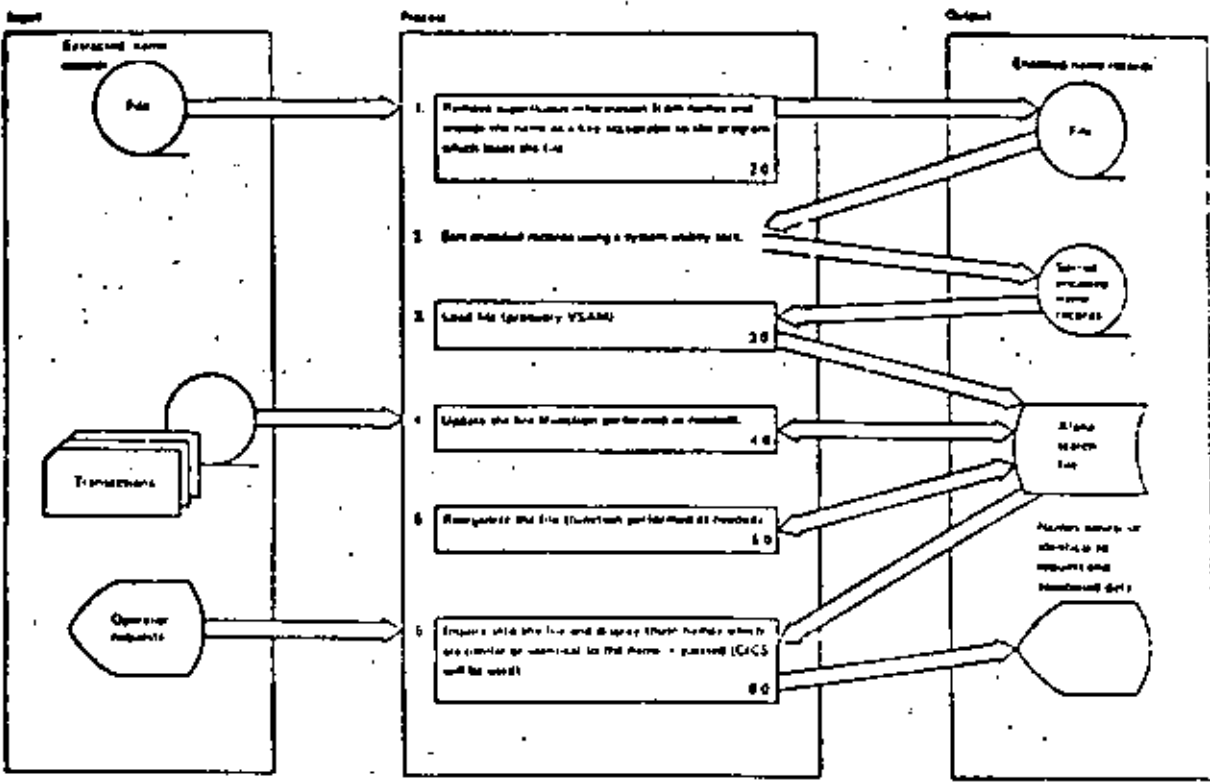


Figure 1.0 Alpha Search Inquiry System Overview Diagram (1.0)

BIBLIOTECAS DE SOPORTE

OBJETIVO:

IDENTIFICAR Y CATALOGAR RUTINAS TIPO PAPA
 USO POSTERIOR.

VENTAJAS:

- MINIMIZA LA MULTIPLICACION DE ESFUERZOS.
- FACILITA LA COMPRESION DE OTROS PROGRAMAS
- AUMENTA LA CAPACIDAD DE DESARROLLO.
- AUMENTA LA CONFIANZA EN EL NUEVO PROGRAMA.

RESUMEN DE RECOMENDACIONES EN PROGRAMACION ESTRUCTURADA
PARA BASIC TRS 80

- AGRUPAR EL MAXIMO DE INSTRUCCIONES POR LINEA FISICA
- COLOCAR SOLO UNA INSTRUCCION POR LINEA LOGICA
- UTILIZAR SOLO ESTRUCTURAS LOGICAS (RECUERDE: EL GØ TØ NUNCA ES NECESARIO)
- USAR LAS CONVENCIONES DE CODIFICACION
- USAR SANGRIA EN FORMA ESTRICTA
- NO CODIFICAR MAS DE TRES NIVELES DE LOGICA SIMULTANEOS
- LIMITAR LOGICAS COMPLETAS A UNA PANTALLA (26 LINEAS)
- COLOCAR DATA AL FINAL
- INCLUIR COMENTARIOS REM Ø
- UTILIZAR NOMBRES DE VARIABLES LOGICOS DE 2 CARACTERES (CUIDADO PALABRAS RESERVADAS DENTRO)
- EXCLUIR EL USO DEL GØ TØ
- ACOPLAR ESTAS PRACTICAS CON PSEUDOCODIGO, SEGMENTACION, DESARROLLO DESCENDENTE, HIPØ Y LAS RECOMENDACIONES PROPIAS DEL LEVEL II
- RECOMENDACION FINAL
- ADOPTA 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 BASICO

SISTEMA DIRECTORIO

SEPTIEMBRE, 1983

①

ESPECIFICACIONES

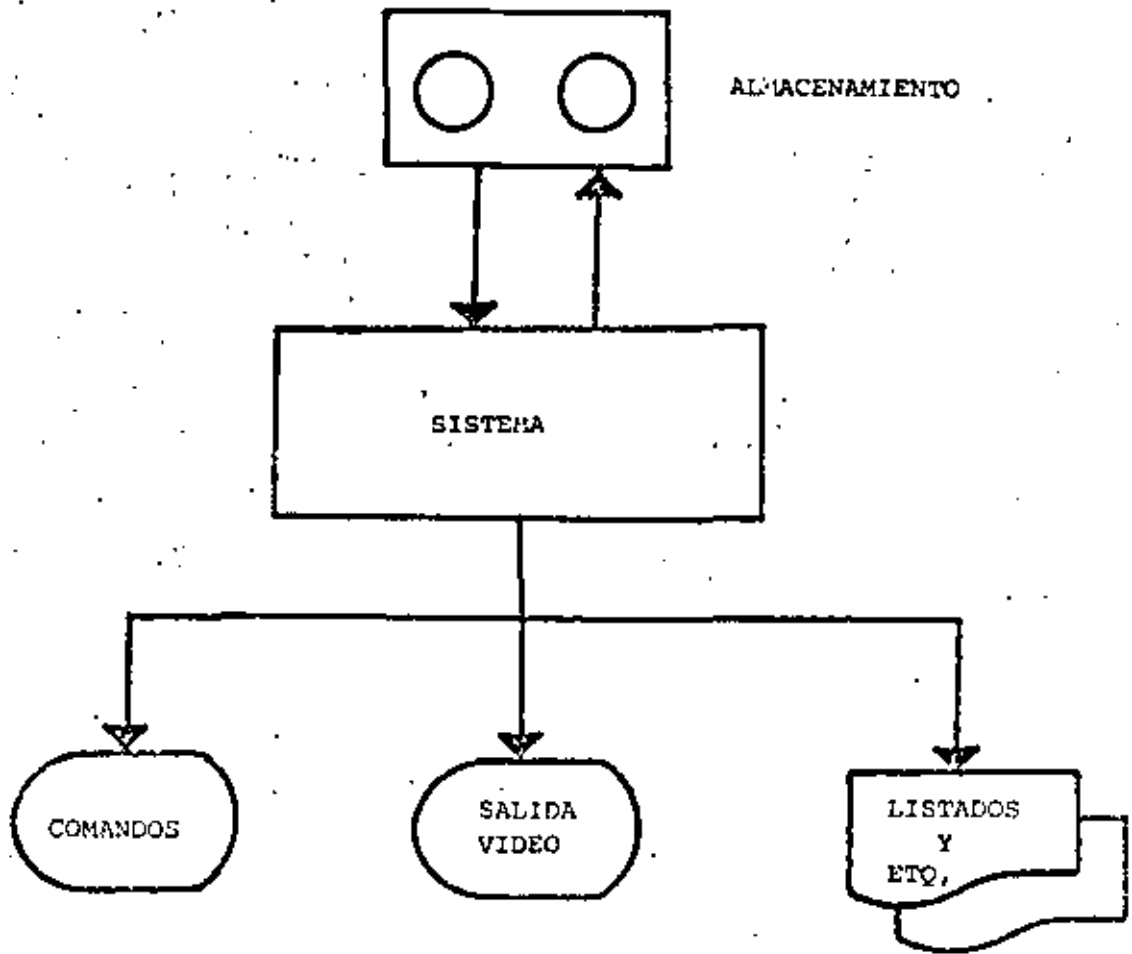
-MANEJAR AUTOMATIZADAMENTE NOMBRES , DOMICILIOS Y DOS SALDOS , TANTO DE CLIENTES COMO DE PROVEEDORES , EN UN DIRECTORIO ACTUALIZABLE CON LAS SIGUIENTES CARACTERISTICAS :

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 "

1							
2	2	3	4	5	6	7	8
3							
4							
5							
6							
7							
8							
9							
10							
11							

(2)

NUMERO	VARIABLE ASOCIADA	TIPO	(LONGITUD)	COMENTARIOS
1	CD	N	—	Número de albaranes en el directorio
2	MX\$ (1)	A	3	llave de acceso valores administrativos C00 - C99 y P00 - P99
3	MN\$ (1)	A	32	Nombre
4	MD\$ (1)	A	40	Dirección
5	DC\$ (1)	A	15	Colonia
6	MP (1)	N	—	Código Postal
7	M1 (1)	N	—	Saldo 1
8	M2 (1)	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

6

-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 "FI\$" 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.

INDICACIONES

-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".

EJEMPLO : P03 , P33 , P00 etc.

-PARA "CARGAR" EL SISTEMA A MEMORIA : PONER LA GRABADORA EN
"PLAY" CON VOLUMEN = 5 Y TECLEAR "LOAD"A".

LISTA DE VARIABLES

(2)

CI si CI=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)
 DLS Delegación (nombre)
 EIS, ENS encabezados
 FIS contiene el formato para la impresión de saldos.
 ID si OK=1, contiene el subíndice donde se encontró (búsqueda).
 IN contiene la opción a ejecutarse.
 KBS 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.
 NDS(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.
 SNS variable para captar respuesta SI o NO.
 VI si VI=1, salida a video.
 X variable para esperar un ENTER.
 X15, X25 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 :

01	TK\$
02(i)	TNS
03(i)	TDS
04	TC\$
05	TP
I	T1
J	T2
MU\$	

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

5290

\$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

(P)

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
12790 XOCHIMILCO

\$ 23
\$ 43

CLAVE : C01
JOSE RICARDO CIRIA MENCE
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 : D47
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
IXCATEDPAN NO. 62
COLONIA LETRAN VALLE
3650 BENITO JUAREZ

CLAVE : P43
CARLOS RAMOS LARIOS
DIRECCION GENERAL DE PROVEDURIA
COLONIA CIUDAD UNIVERSITARIA
4960 COYOACAN

CLAVE : P75
PROVEEDORA DE PAPELERIA ESPECIAL S.A.
VIENA NO. 133
COLONIA PROGRESO
5400 CUAJIMALPA

CLAVE : P77
HERIBERTO OLGUIN ROMO
RINCON DE LOS ABETOS 101
COLONIA RINCONADA COAPA
15780 XOCHIMILCO

15 JOSE RICARDO CIRIA MERCE 1983

20 COSUB 3001

COSUB 3201

COSUB 180

50 IF IN=1 THEN GOTO 160

40 COSUB 3201

COSUB 3501

COSUB 3701

ON IN GOTO 70,80,90,100,110,120,130,140

70 COSUB 2101

GOTO 150 1-SALIDA DEL SISTEMA

80 COSUB 4201

GOTO 150 2-ALTA

90 COSUB 21201

GOTO 150 3-BAJA

100 COSUB 19801

GOTO 150 4-CAMBIO

110 COSUB 13001

GOTO 150 5-MUESTRA PANTALLA

120 COSUB 13801

GOTO 150 6-LISTA

130 COSUB 14001

GOTO 150 7-ETIQUETAS

140 COSUB 20801

GOTO 150 8-BUSCA

150 GOTO 501

END CASE

160 END WHILE

170 COSUB 3201

PRINT "FIN DE LA EJECUCION":

END

180 REM SUBROUTINA INICIO

190 INPUT "NUEVO DIRECTORIO (SI,NO)";SN\$:

IF SN\$ <> "SI" THEN:

COSUB 1110

ELSE

CL=1:

END IF

200 RETURN

300 REM SUBROUTINA INICIALIZA

305 DIM NO\$(16):

COSUB 940

310 EN\$="O.E.C.F.I. SISTEMA DIRECTORIO":

SI\$="FACULTAD DE INGENIERIA U. N. A. M.":

Z=0:

IX=10:

FI\$="#####":

RETURN

320 REM SUBROUTINA ENCABEZADO

330 CLS:

PRINT STRING\$(10,"*");E!\$:

PRINT STRING\$(10,"*");EN\$:

PRINT

210 REM SUBROUTINA TERMINA

220 COSUB 320:

IF C1=1 THEN

COSUB 1150

NOELSE

END IF

230 RETURN

340 REM SUBROUTINA 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 SUBROUTINA 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 SUBROUTINA 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 XOCOTLILCO

960 REM SUBROUTINA ESPERA ENTER
970 PRINT*PARA CONTINUAR OPRIMA < ENTER >*;
INPUT X
RETURN

910 REM SUBROUTINA OBTIENE DELEGACION
920 IF (DE < 1) OR (DE > 16) THEN
DL\$=STRING\$(10, ' ');
ELSE
DL\$=ND\$(DE)
ENDIF
930 RETURN

2160 REM SUBROUTINA PARAMETROS LISTA TODD
2170 X1\$="C";
X2\$="P";
RETURN

1110 REM SUBROUTINA CARGA DIRECTORIO
1120 PRINT*DE <-REWIND > AL CASSETTE Y*;
PRINT*PONCALO EN < PLAY > CON VOLUMEN #5*;
GOSUB 950
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 SUBROUTINA GUARDA DIRECTORIO
1160 PRINT*DE REWIND AL CASSETTE Y*;
PRINT*PONCALO EN RECORD*;
GOSUB 950
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

```

GOSUB 3201
PRINT '>>>>CAMBIO<<<<'
GOSUB 5201
KB=OK
GOSUB 1650
2000 IF OK <> 1 THEN
PRINT 'NO EXISTE LA CLAVE : 'OK
ELSE
LI=ID
LB=ID
VI=1
PR=0
GOSUB 660
GOSUB 1880
'END IF
2010 GOSUB 960
RETURN

```

```

1840 REM-SUBROUTINA PREGUNTA CAMBIO
1850 INPUT 'DESEA CAMBIARLO (SI,NO) 'SN
CA=0
1860 IF SN='SI' THEN
CA=1
'NOELSE
ENDIF
1870 RETURN

```

```

1880 REM SUBROUTINA CAMBIA LLAVE
1890 PRINT 'NOMBRE: ',MN(ID):
GOSUB 1840
1895 IF CA THEN
GOSUB 590
MN(ID)=ON:
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

```

```

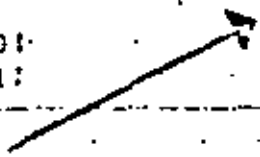
1940 PRINT 'CODIGO POSTAL:',MP(ID):
GOSUB 1840
1950 IF CA THEN
GOSUB 790
MP(ID)=DP:
C1=1
'NOELSE
END IF
1960 PRINT 'SALDO 1 : ',M1(ID):
GOSUB 1840
1962 IF CA THEN
GOSUB 820
M1(ID)=D1:
C1=1
'NOELSE
END IF

```

```

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 SUBROUTINA ALTA
430 GOSUB 320:
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 SUBROUTINA VERIFICA ESPACIO DISPONIBLE
470 IF CD >= MX THEN
PRINT 'TABLA DEL DIRECTORIO LLENA':
ELSE
GOSUB 490:
ENDIF
480 RETURN

```

```

490 REM SUBROUTINA ADICIONA CLAVE
500 C1=1:
CD=CD+1:
MK$(CD)=DK$:
GOSUB 590:
MN$(CD)=DN$:
GOSUB 650:
MD$(CD)=DO$:
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 SUBROUTINA BUSCA LLAVE
1660 OK=0
      NO=0
1670 IF CD=0 THEN
      OK=0
      ELSE
      GOSUB 1690
      'END IF
1680 RETURN

1690 REM SUBROUTINA ITERA BUSQUEDA
1700 LI=1:
      LS=CD
1710 IF (OK=1) OR (NO=1) THEN 1730
1720 GOSUB 1740:
      GOTO 1710:
      'END DO
1730 RETURN

1740 REM SUBROUTINA BUSCA
1750 IF (LI+1)=LS OR (LI=LS) THEN
      GOSUB 1770
      ELSE
      GOSUB 1800
      'END IF
1760 RETURN

1770 REM SUBROUTINA 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 SUBROUTINA 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 SUBROUTINA 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 SUBROUTINA 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 SUBROUTINA 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 SUBROUTINA 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 SUBROUTINA CAPTA CODIGO
800 INPUT "CODIGO POSTAL " ;DP$
810 RETURN
```

```
820 REM SUBROUTINA CAPTA SALDO 1
830 INPUT "SALDO 1 " ;D1$
RETURN
```

```
840 REM SUBROUTINA CAPTA SALDO 2
850 INPUT "SALDO 2 " ;D2$
RETURN
```

```

1450 REM SUBROUTINA ORDENA ( QUICK-SORT )
1460 O1=1:
      O2(O1)=1:
      O3(O1)=O
1470 IF O1 <= 0 THEN GOTO 1610
1480 O4=O2(O1):
      O5=O3(O1):
      O1=O1-1
1490 IF O4 >= O5 THEN GOTO 1600
1500 I=O4:
      J=O5:
      MU$=MK$(INT((O4+O5)/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 < O5 THEN
      O1=O1+1:
      O2(O1)=I:
      O3(O1)=O5:
      'NOELSE
      ENDIF
1590 O5=J:
      GOTO 1490:
      'END DO
1600 GOTO 1470:
      'END DO
1610 RETURN

```

```

1020 REM SUBROUTINA INTERCAMBIA I-J
1030 TK$=MK$(I) : MK$(I)=MK$(J) : MK$(J)=TK$
1040 TN$=KN$(I) : KN$(I)=KN$(J) : KN$(J)=TN$
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) : HP (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 SUBROUTINA 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

```

```

1230 REM SUBROUTINA VIDEO
1235 CLS
1240 PRINT "CLAVE : " ; MK$(I)
      PRINT MN$(I)
      PRINT MD$(I)
      PRINT "COLONIA : " ; MC$(I)
      PRINT MP(I) ; " " ; DL$
      PRINT USING F1$ ; M1(I)
      PRINT USING F1$ ; M2(I)
1245 GOSUB 960
1250 RETURN

```

```

1200 REM SUBROUTINA SALIDA
1210 IF VI=1
      GOSUB 1230
    ELSE
      GOSUB 1260
    END IF
1220 RETURN

```

```

1260 REM SUBROUTINA PAPEL Y ETIQUETA
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 SUBROUTINA MUESTRA ETIQUETAS
1410 VI=0
      PR=0
      PRINT "MONTAR ETIQUETAS"
      GOSUB 960
      GOSUB 1320
      RETURN

```

```

1380 REM SUBROUTINA MUESTRA PAPEL
1390 VI=0
      PR=1
      PRINT "MONTAR PAPEL DIRECTORIO"
      GOSUB 960
      GOSUB 1320
      RETURN

```

```

1300 REM SUBROUTINA MUESTRA VIDEO
1310 VI=1
      PR=0
      GOSUB 1320
      RETURN

```

```

1320 REM SUBROUTINA 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 SUBROUTINA BAJA
2130 GOSUB 21601
GOSUB 3201
PRINT ">>>>> BAJA <<<<<";
GOSUB 5201
KR$=DK$;
GOSUB 1650
2140 IF OK <> 1 THEN
PRINT "NO EXISTE LA CLAVE ";DK$;
GOSUB 960
ELSE
LI=ID;
LS=ID;
VI=1;
PR=0;
GOSUB 860;
GOSUB 980;
END IF
2150 RETURN

```

```

980 REM SUBROUTINA 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;
INELSE
ENDIF
1010 RETURN

```

```

2080 REM SUBROUTINA BUSCA (MENU)
2090 GOSUB 21601
GOSUB 3201
GOSUB 5201
KR$=DK$;
GOSUB 1650
2100 IF OK <> 1 THEN
PRINT "NO EXISTE LA CLAVE ";DK$;
GOSUB 960
ELSE
LI=ID;
LS=ID;
VI=1;
PR=0;
GOSUB 860;
END IF
2110 RETURN

```


1 BYTE

HEX:	43	55	52	53	4F	20	44	45	3A	42	41	53	49	43
DEC:	67	85	82	83	79	32	68	69	58	66	65	83	73	67
	C	U	R	S	O		D	E	:	B	A	S	I	C

- 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

A1\$ = "ABCDE"

A2\$ = "12.45"

A3\$ = "56"

A4\$ = "78"

A5\$ = "-78"

N1 = -54.21

N = 54.21

CONCATENACION

A1\$ + A2\$ _____ ABCDE12.45

A4\$ + A5\$ _____ 7856

LONGITUD

LEN (A1\$) _____ 5

LEN (A5\$) _____ 3

CONVERSION STRING -> NUMERICO

VAL (A2\$) _____ 12.45

VAL (A3\$ + "." + A4\$) _____ 56.78

VAL ("150 PESOS") _____ 150

VAL (A4\$ + "E" + "15") _____ 7.8E+16

CONVERSION NUMERICO -> STRING

STR\$ (N) _____ 54.21

STR\$ (N - N1) _____ 0

STR\$ (N) + " PESOS " _____ 54.21 PESOS

STRING DE ELEMENTOS IGUALES

STRING \$ (10, "**") _____ *****

STRING \$ (5, 42) _____ *****

SUB-STRING (IZQUIERDA)

LEFT \$ (A1\$, 2) _____ AB

LEFT \$ (A1\$, 10) _____ ABCDE

SUB-STRING (DERECHA)

RIGHT (A1\$, 2) _____ DE

RIGHT (A1\$, 10) _____ ABCDE

SUB-STRING (CENTRO)

MID \$ (A1\$, 3, 1) _____ C

MID \$ (A1\$, 3) _____ CDE

CONVERSION CARATER -> ASCII

ASC (A1\$) _____ 65

ASC (RIGHT\$ (A2\$, 1)) _____ 53

CONVERSION ASCII -> CARACTER :

CHR\$ (34) + "HOLA" + CHR\$ (34) _____ "HOLA"

OTRAS

FRE (A1\$) = FRE (L\$) _____ 40

TIME \$ _____ 03/04/03 15:01:55

INKEY\$ _____

```

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   CO=CO+1:
   PRINT "ACUMULADO : *; CO :
   A$=INKEY$:
   GOTO 20:
   'END DO
40 PRINT "ACERTASTE ... ES UNA *; CHR$(34); 'A' ; CHR$(34):
   END

```

A1\$ = "ABCDE"	A2\$ = "12.45"	A3\$ = "56"
A4\$ = "78"	A5\$ = "-78"	N1 = -54.21
M = 54.21		

CONCATENACION

A1\$ + A2\$	ABCDE12.45
A4\$ + A3\$	7856

LONGITUD

LEN (A1\$)	5
LEN (A5\$)	3

CONVERSION STRING -> NUMERICO

VAL (A2\$)	12.45
VAL (A3\$ + "." + A4\$)	56.78
VAL ("150 PESOS")	150
VAL (A4\$ + "E" + "15")	7.8E+16

CONVERSION NUMERICO -> STRING

STR\$ (M)	54.21
STR\$ (M - N1)	0
STR\$ (M) + " PESOS "	54.21 PESOS

STRING DE ELEMENTOS IGUALES

STRING \$ (10,"*")	*****
STRING \$ (5,42)	*****

SUB-STRING (IZQUIERDA)

LEFT \$ (A1\$,2)	AB
LEFT \$ (A1\$,10)	ABCDE

SUB-STRING (DERECHA)

RIGHT (A1\$,2)	DE
RIGHT (A1\$,10)	ABCDE

SUB-STRING (CENTRO)

MID \$ (A1\$,3,1)	C
MID \$ (A1\$,3)	CDE

CONVERSION CARACTER -> ASCII

ASC (A1\$)	65
ASC (RIGHT\$(A2\$,1))	53

CONVERSION ASCII -> CARACTER :

CHR\$(34) + "DOL." + CHR\$(34)	"10LA"
--------------------------------	--------

OPUS

FRE (A1\$) + FRE (B\$)	40
TIME \$	03/04/87 15:01:55
INQUI	

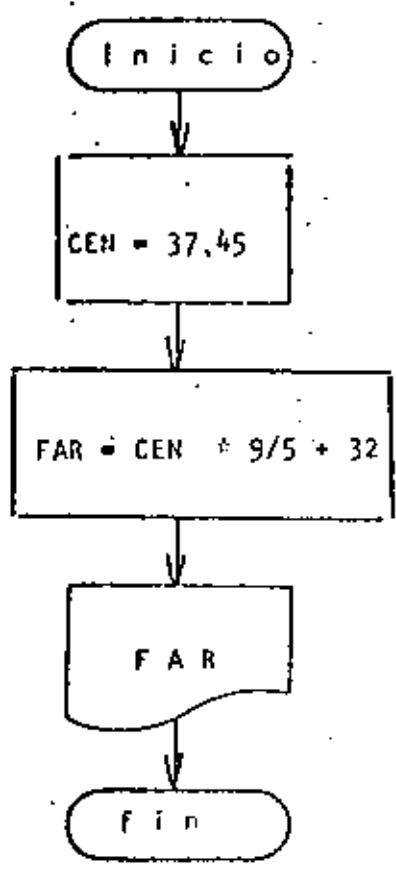


DIVISION DE EDUCACION CONTINUA
FACULTAD DE INGENIERIA U.N.A.M.

LENGUAJE DE PROGRAMACION BASIC

EJEMPLOS

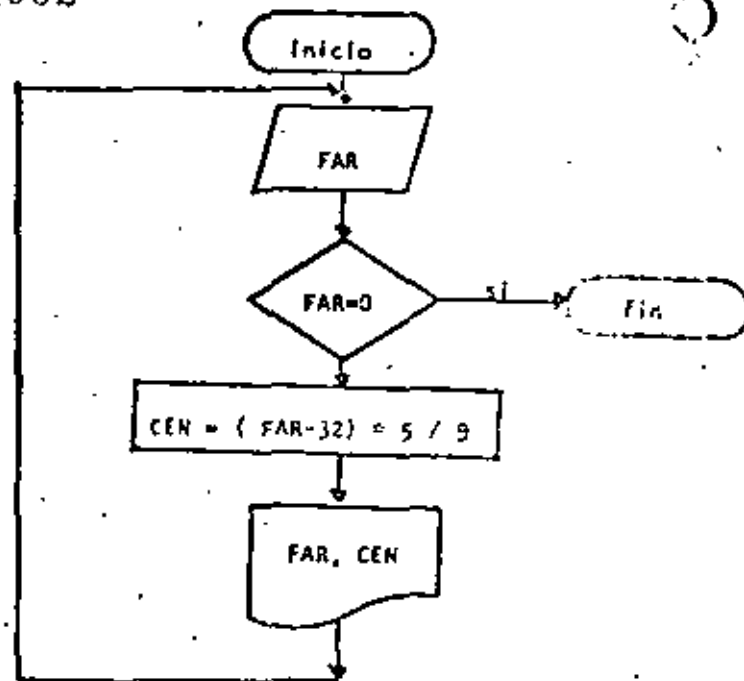
SEPTIEMBRE, 1983



```
1  REM ***** (10) *****  
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

1002



```

10 REM-----DOS-----
20 REM CONVERSION DE GRADOS FARENHEIT A GRADOS CENTIGRADOS
30 REM
40 PRINT "CONVERSION DE GRADOS FARENHEIT A GRADOS CENTIGRADOS"
50 PRINT
60 "INGRESA 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 "END"
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? 55

55 GRADOS FARENHEIT SON 13.3333 GRADOS CENTIGRADOS

FARENHEIT? -95

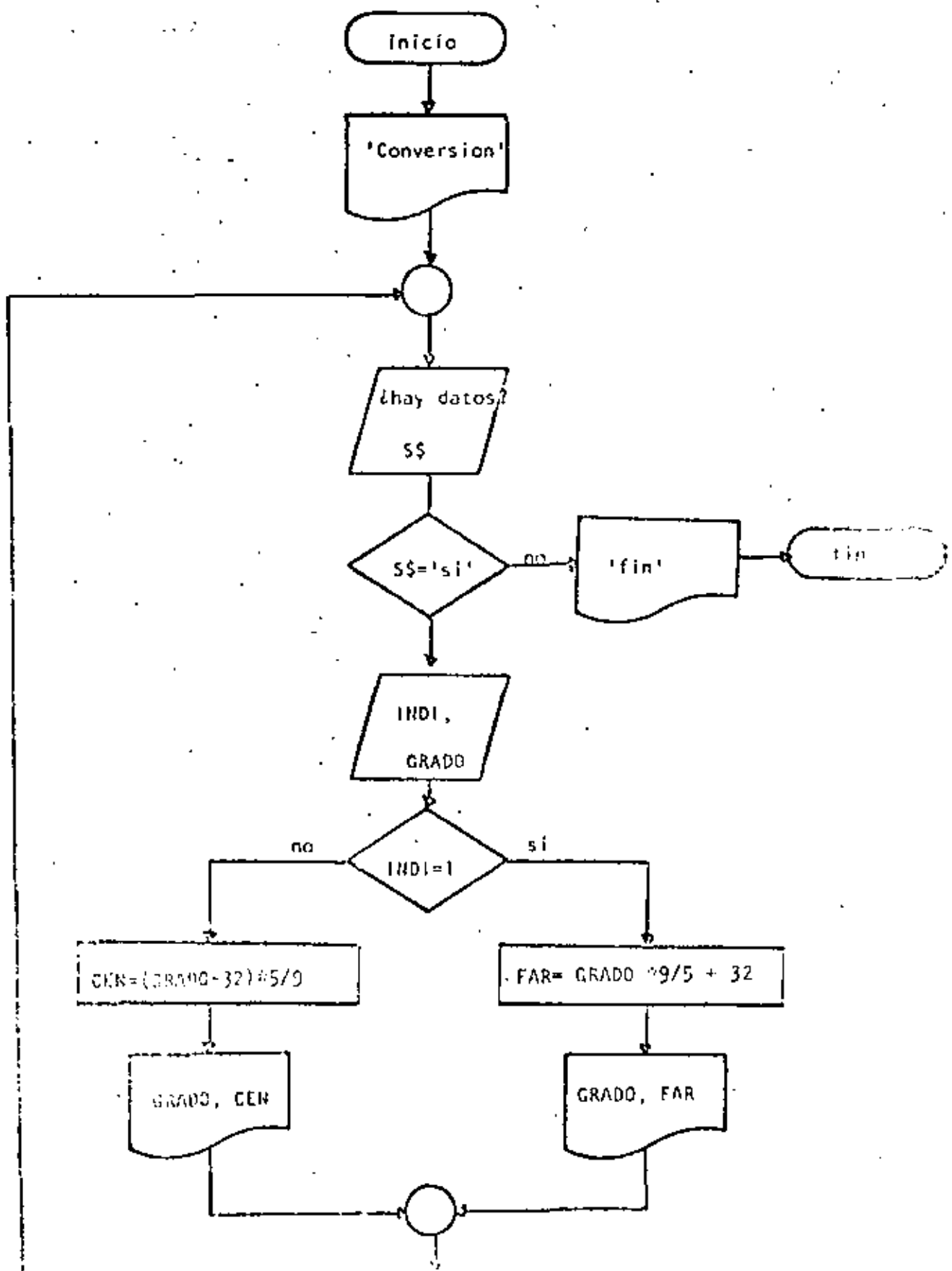
-95 GRADOS FARENHEIT SON -42.7778 GRADOS CENTIGRADOS

FARENHEIT? 0

FIN DEL PROGRAMA

Ready

CONVERSION ENTRE GRADOS CENTIGRADOS Y GRADOS FARENHEIT




```

10 REM-----TRES-----
  1 REM CONVERSION ENTRE GRADOS CENTIGRADOS Y GRADOS FARENHEIT
30 REM
40 PRINT "CONVERSION DE GRADOS CENTIGRADOS A FARENHEIT O VICEVERSA"
50 PRINT
60 'DOWHILE PAYA DATOS
70 INPUT "HAY DATOS (SI O NO)?" :SI
80 IF SI <> "SI" THEN 200
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 30
210 '
220 '
230 PRINT "FIN DEL PROCESO"
240 END

```

Ready

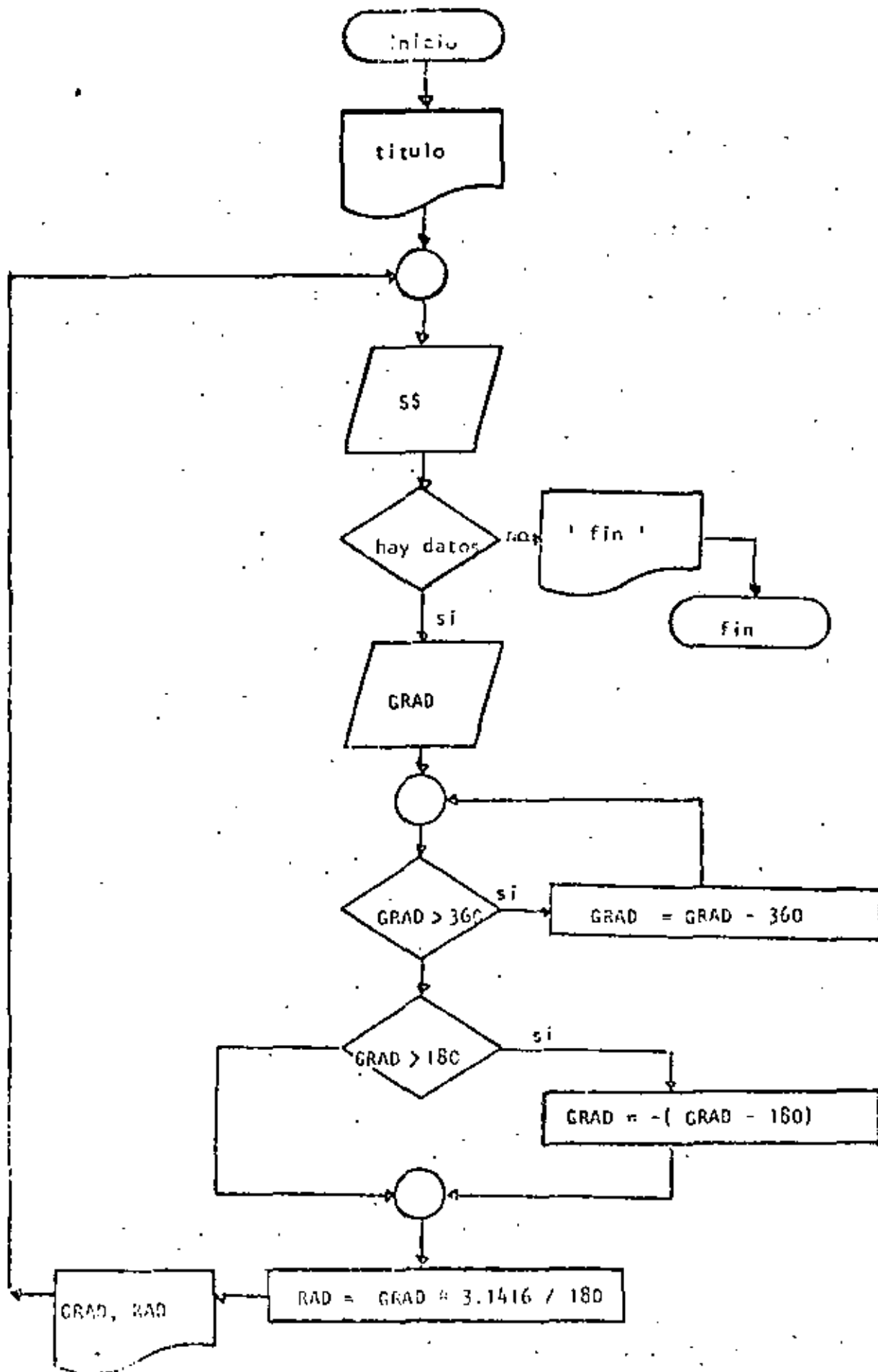
```

>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
45 GRADOS FARENHEIT SON 18.3333 GRADOS CENTIGRADOS
HAY DATOS (SI O NO)? NO
FIN DEL PROCESO

```

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)":SI
70 IF SI <> "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

```

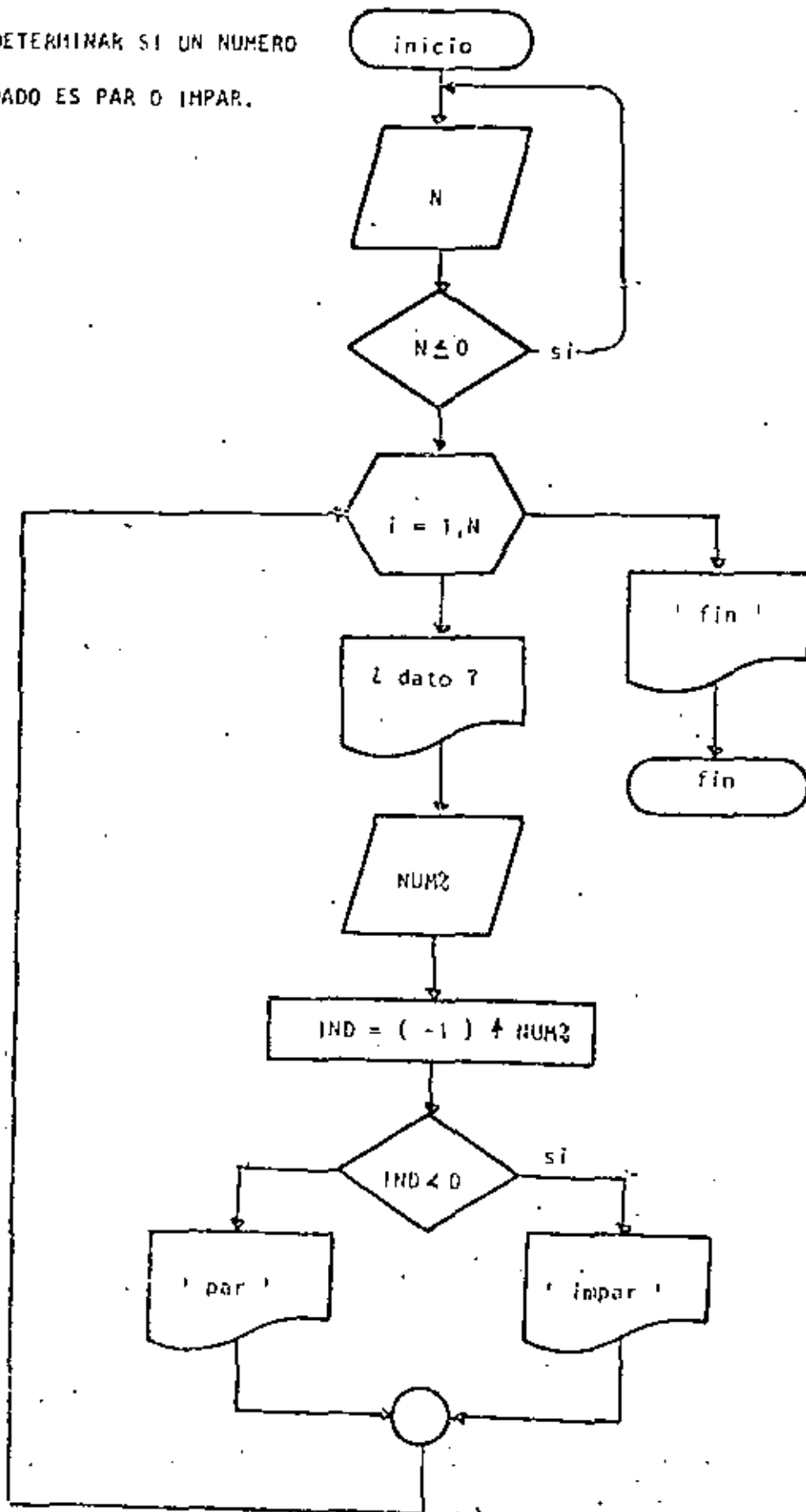
NUM
CONVERSION DE GRADOS A RADIANES

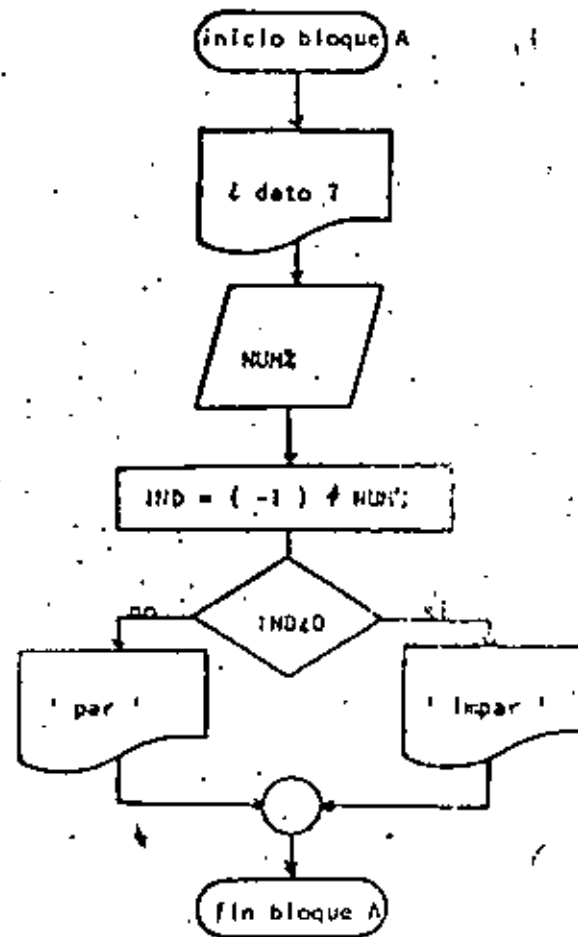
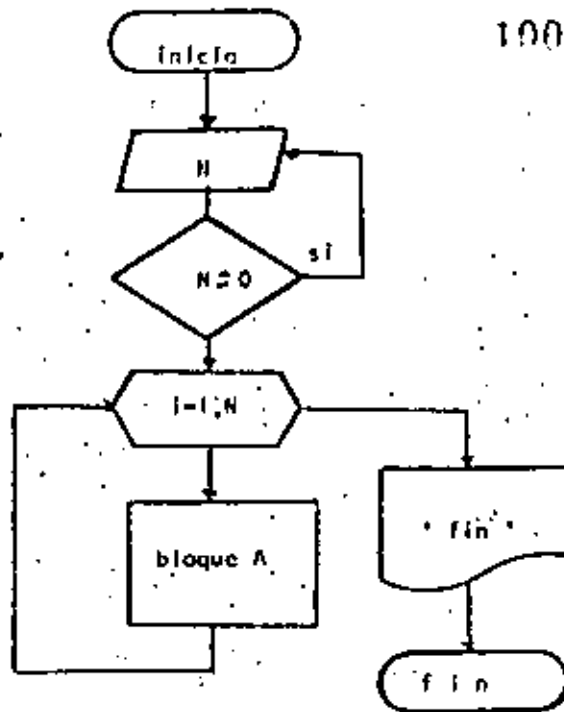
```

HAY DATOS(SI O NO)? SI
GRADOS? 34
34 GRADOS SON .592413 RADIANES
HAY DATOS(SI O NO)? SI
GRADOS? 194
194 GRADOS SON 3.394397 RADIANES
HAY DATOS(SI O NO)? SI
GRADOS? 157
157 GRADOS SON 2.7129 RADIANES
HAY DATOS(SI O NO)? SI
GRADOS? 170
170 GRADOS SON 2.9644 RADIANES
HAY DATOS(SI O NO)? NO
FIN DEL PROCESO
Ready

```

DETERMINAR SI UN NUMERO
DADO ES PAR O IMPAR.



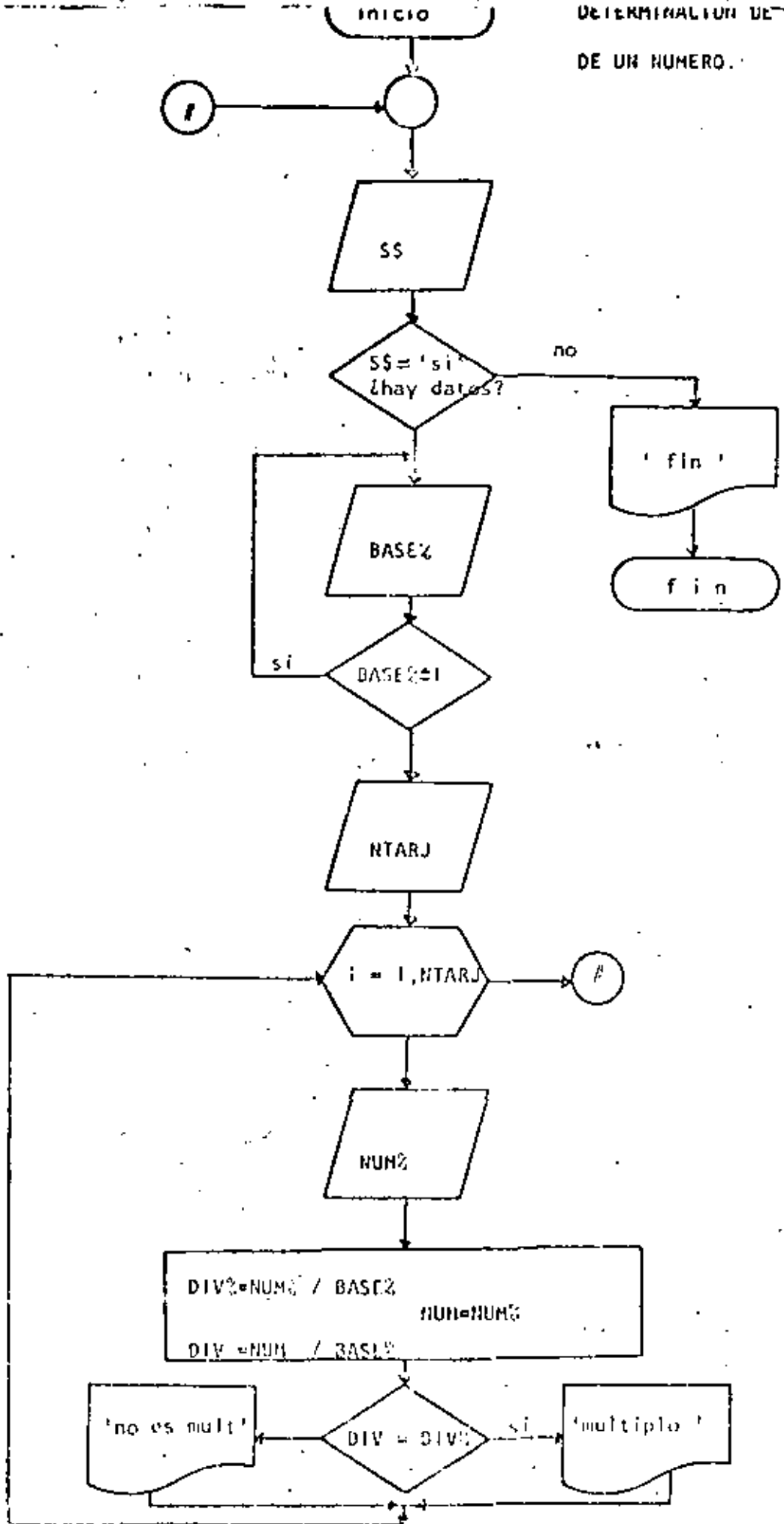


```

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 GO
50 FOR J=1 TO N
60     PRINT "DAME EL DATO NUMERO:" J
70     INPUT NUMZ
80     INDZ=(-1)*NUMZ
90     IF INDZ < 0 THEN PRINT "EL NUMERO " NUMZ " ES IMPAR "
        ELSE PRINT "EL NUMERO " NUMZ " ES
        PAR "
100 NEXT J
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 ? 4
EL NUMERO 4 ES PAR
DAME EL DATO NUMERO: 3 ? 07
EL NUMERO 07 ES IMPAR
FIN DEL PROGRAMA
Ready
  
```



```

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)?:SI
60 IF SI <> 'SI' THEN 170
70     INPUT 'BASE:BASEX: IF BASEX <=1 THEN 70
80     INPUT 'CANTIDAD DE DATOS':NTARJ
90     FOR I=1 TO NTARJ
100        PRINT 'DATO NUMERO ':I:INPUT NUMZ
110        DIVZ=NUMZ/BASEX
120        NUM=NUMZ
130        DIV=NUM/BASEX
140        IF DIV=DIVZ THEN PRINT NUMZ:' SI ES MULTIPLO DE BASEX
            ELSE PRINT NUMZ:' NO ES

```

```

MULTIPLO DE :BASEX
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

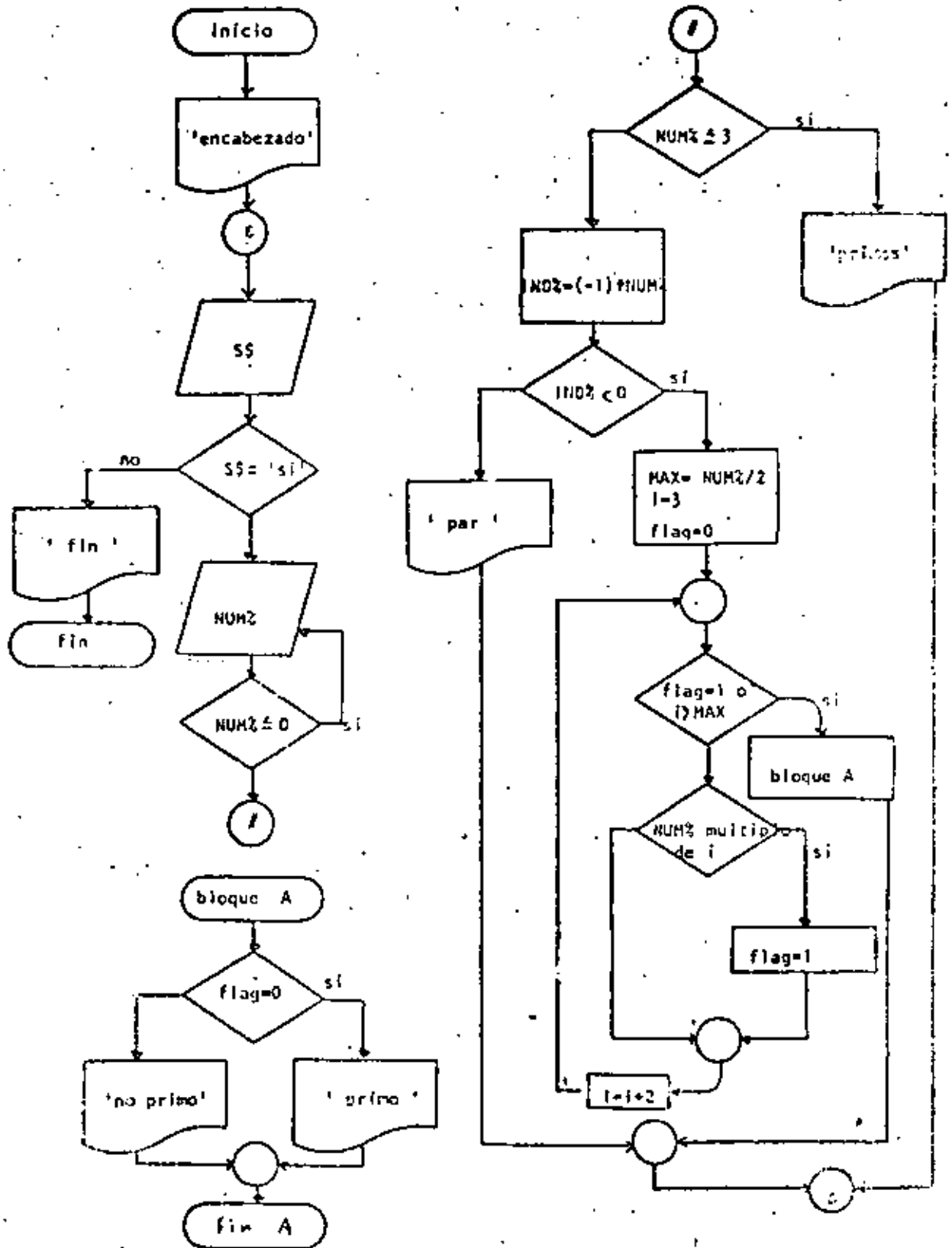
```

```

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 ? 6
6 SI ES MULTIPLO DE 3
HAY DATOS (SI O NO)? NO
FIN DEL PROCESO
Ready

```

1011
 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";NUMZ
80 IF NUMZ <= 0 THEN 70
90 IF NUMZ <= 3 THEN 110
100 GOTO 150
110 'THEN
120 PRINT TAB(42);NUMZ
130 ' 1 2 Y 3 SON PRIMOS
140 GOTO 360
150 'ELSE
160 INDZ=(-1)*NUMZ
170 IF INDZ < 0 THEN 200
180 GOTO 330
190 'THEN
200 'NUMERO IMPAR
210 MAX=NUMZ/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 NUMZ=FIX(NUMZ/I)*I THEN FLAG=1
270 I=I+2
280 GOTO 240
290 'ENDWHILE
300 IF FLAG=0 THEN PRINTTAB(42);NUMZ
ELSE PRINTTAB(52);
NUMZ
310 GOTO 350
320 'ELSE
330 PRINT TAB(52);NUMZ;"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

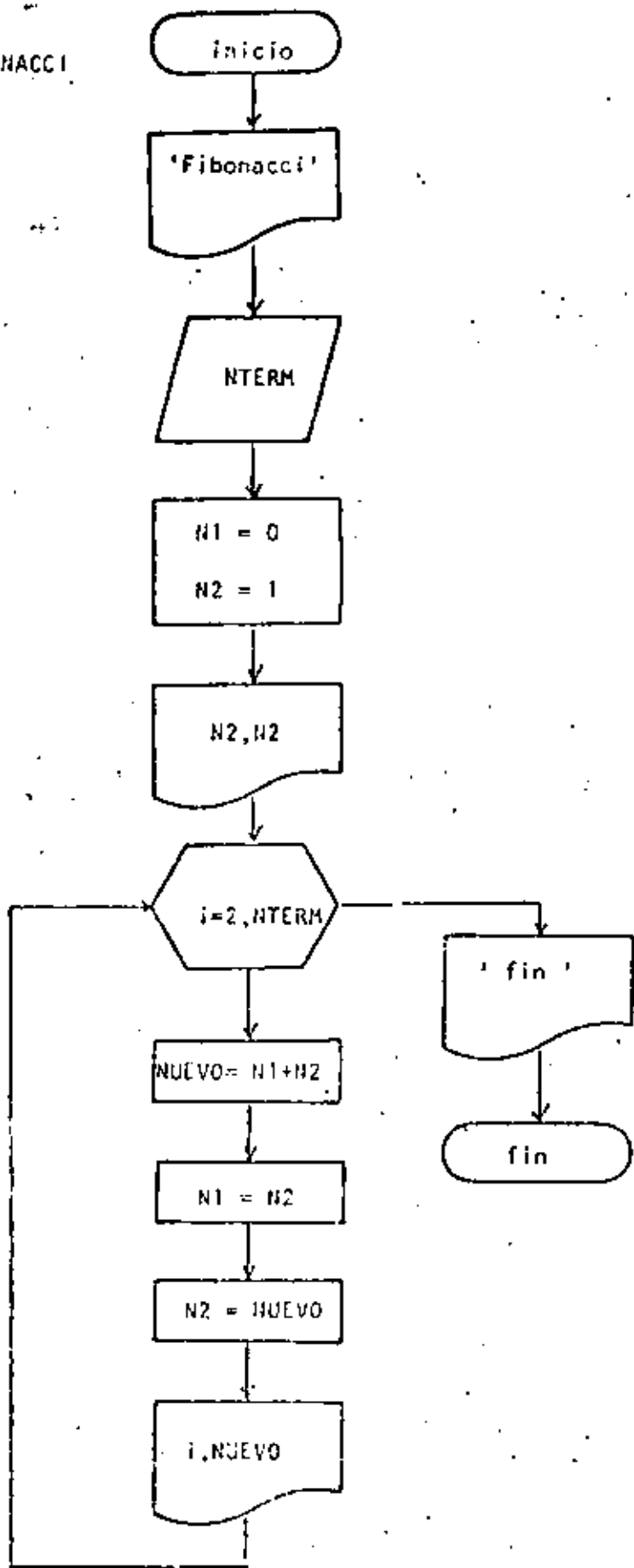
```

>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		

Rearto

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

```

Reads

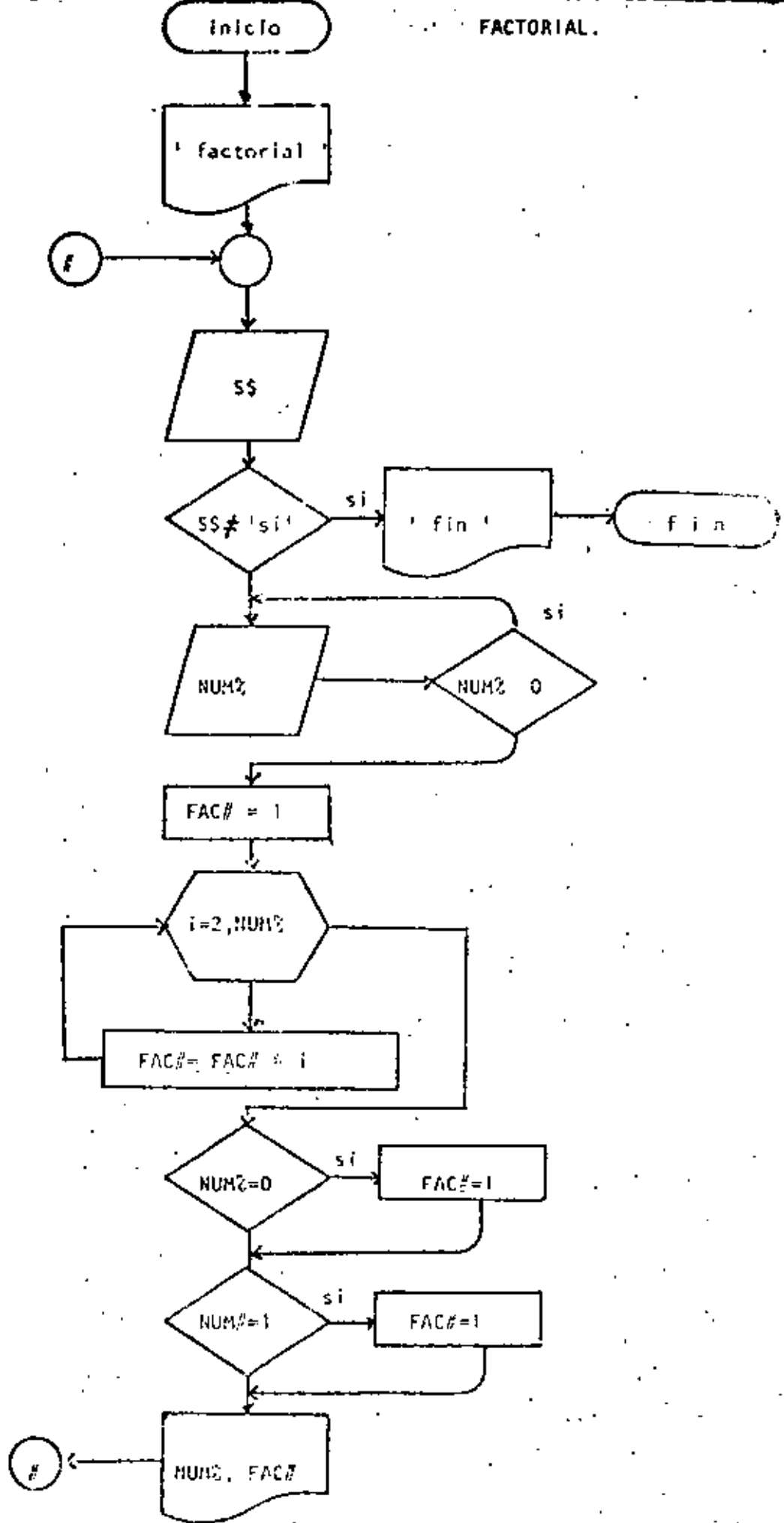
>RUN
SERIE DE FIBONACCI

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

Reads



```

1  REM-----DUEVE-----
10 REM FACTORIAL
20 REM
30 PRINT "FACTORIAL":PRINT
40 'DOWHILE HAYA DATOS
50 INPUT "HAY DATOS (SI O NO)":SI
60 IF SI <> "SI" THEN 180
70     INPUT"DAME EL DATO":NUMZ
80     IF NUMZ < 0 THEN 70
90     FACT#=1
100    FOR I=2 TO NUMZ
110        FACT#=FACT#*I
120    NEXT I
130    IF NUMZ=0 THEN FACT#=1
140    IF NUMZ=1 THEN FACT#=1
150    PRINT "EL FACTORIAL DE " ;NUMZ;" VALE " ;FACT#
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 2.6525205981219110+32
HAY DATOS (SI O NO)? SI
DAME EL DATO? 32
EL FACTORIAL DE 32 VALE 2.6313083673369260+35
HAY DATOS (SI O NO)? SI
DAME EL DATO? 35
?00 I=200 In 110
Ready
>RUN
FACTORIAL

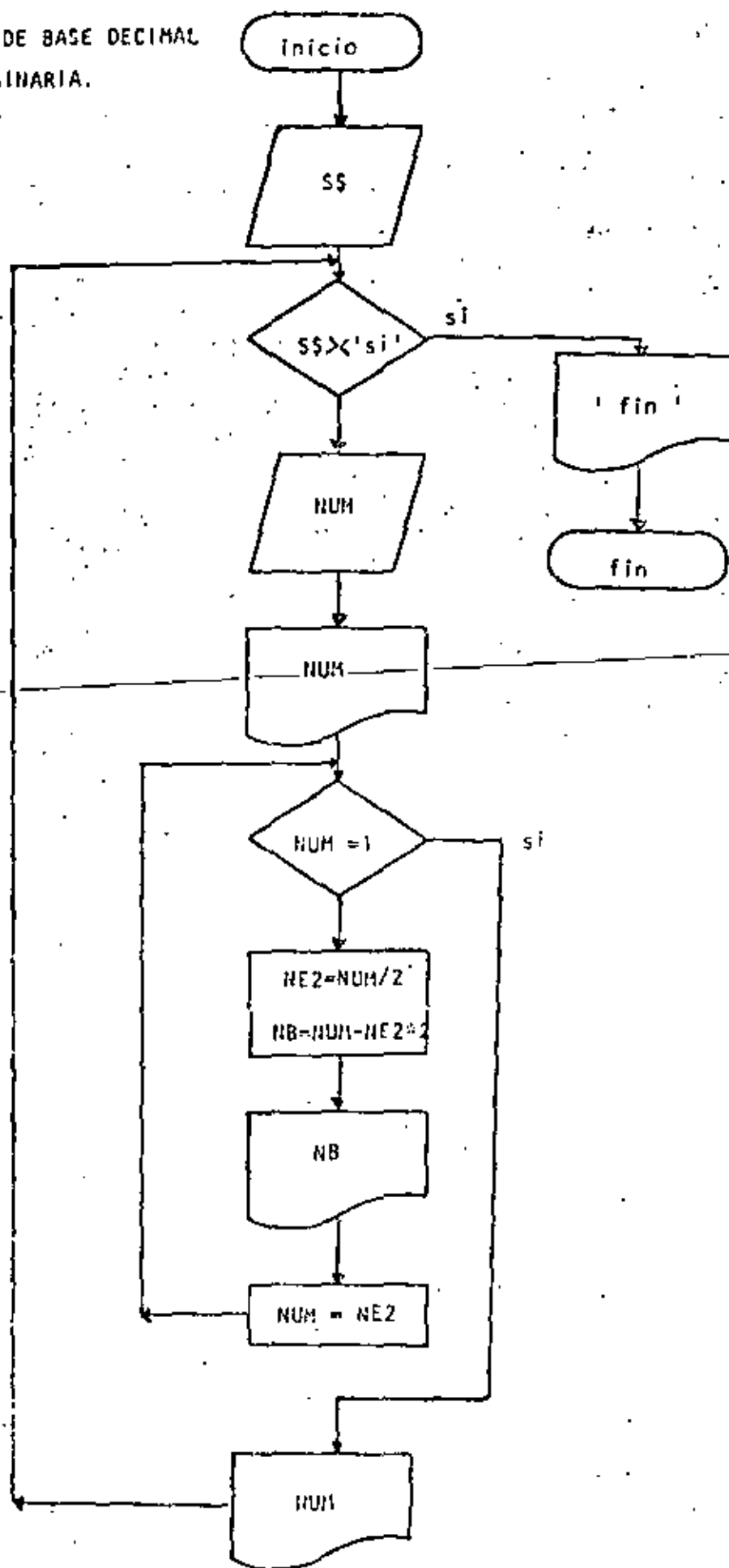
```

```

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)? SI
70 IF SI <> '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         NEZ=NUM/2
130         NE =NUM - NEZ*2
140         PRINT TAB(10);NE
150         NUM=NEZ
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

```

```

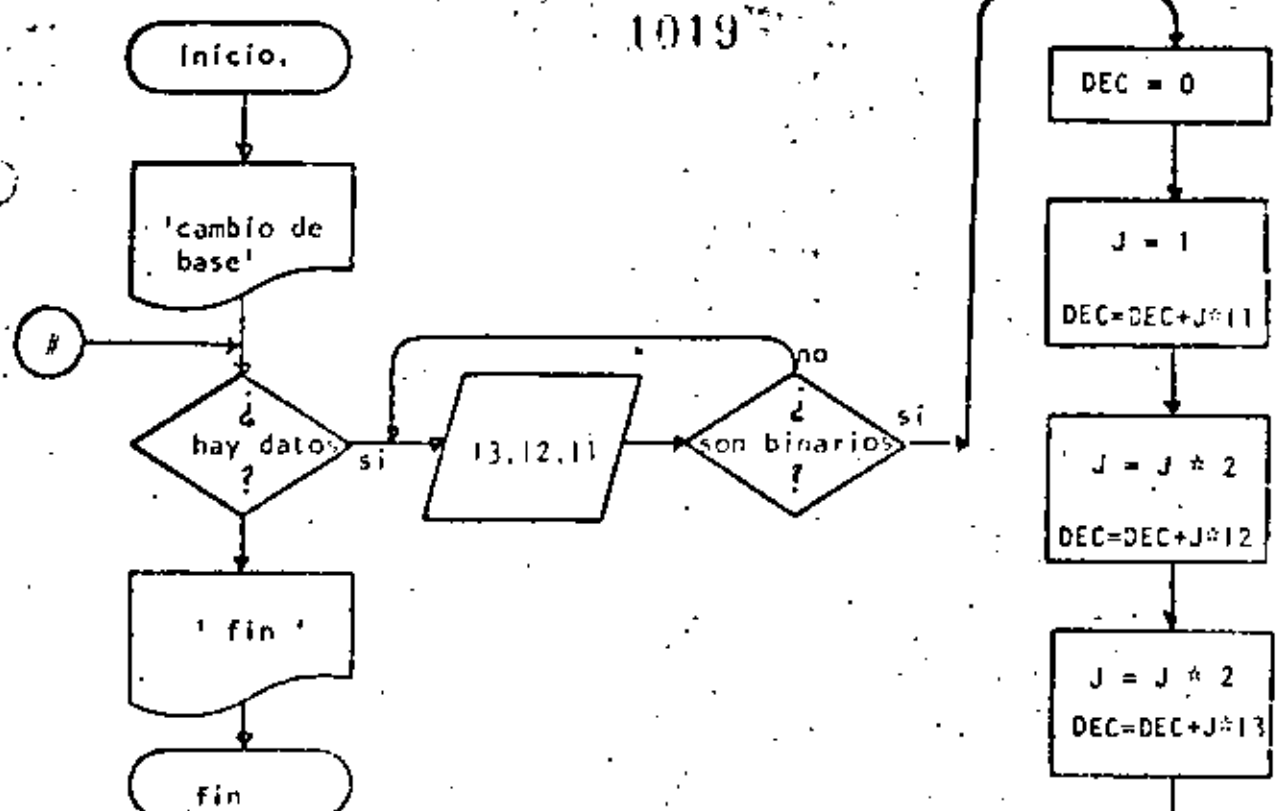
HAY DATOS (SI O NO)? SI
? 4
EL NUMERO 4 EN BASE DECIMAL ES IGUAL A
0
0
1 EN BASE BINARIA

```

```

HAY DATOS (SI O NO)? NO
FIN DEL CAMBIO DE BASES
Kerds

```



```

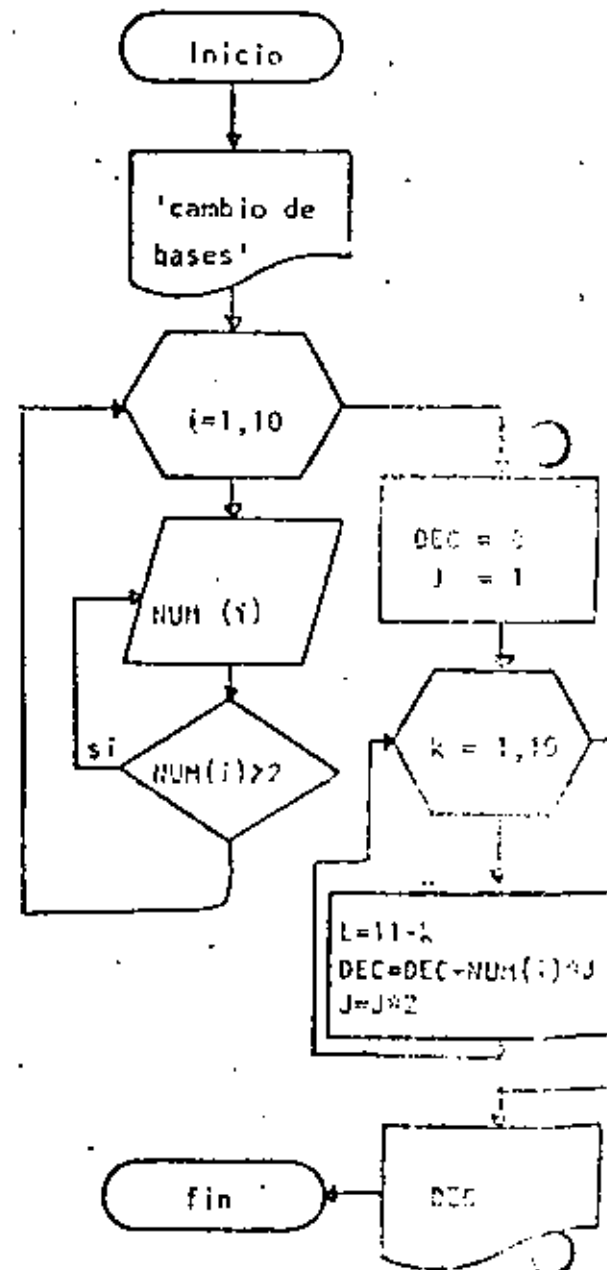
30 REM-----ORCE-----
20 REM CAMBIO DE BASE BINARIA A BASE DECIMAL
30 REM
40 PRINT "CAMBIO DE BASE BINARIA A BASE DECIMAL"
50 REM SI HAYA DATOS
60 INPUT "¿ HAY DATOS (SI O NO)? "; S1: IF S1 <> "SI" THEN 190
70 INPUT " LOS TRES DIGITOS BINARIOS "; I1,I2,I3
80 IF ASC(I1)>1 OR ASC(I2)>1 OR ASC(I3)>1 THEN 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 "; I1;I2;I3; " CORRESPONDE AL ";DEC;" EN BASE DECIMAL"
170 GOTO 40
180 END
190 PRINT "FIN DE CAMBIO DE BASES"
200 END

```

CAMBIO DE BASE BINARIA A BASE DECIMAL
¿ HAY 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

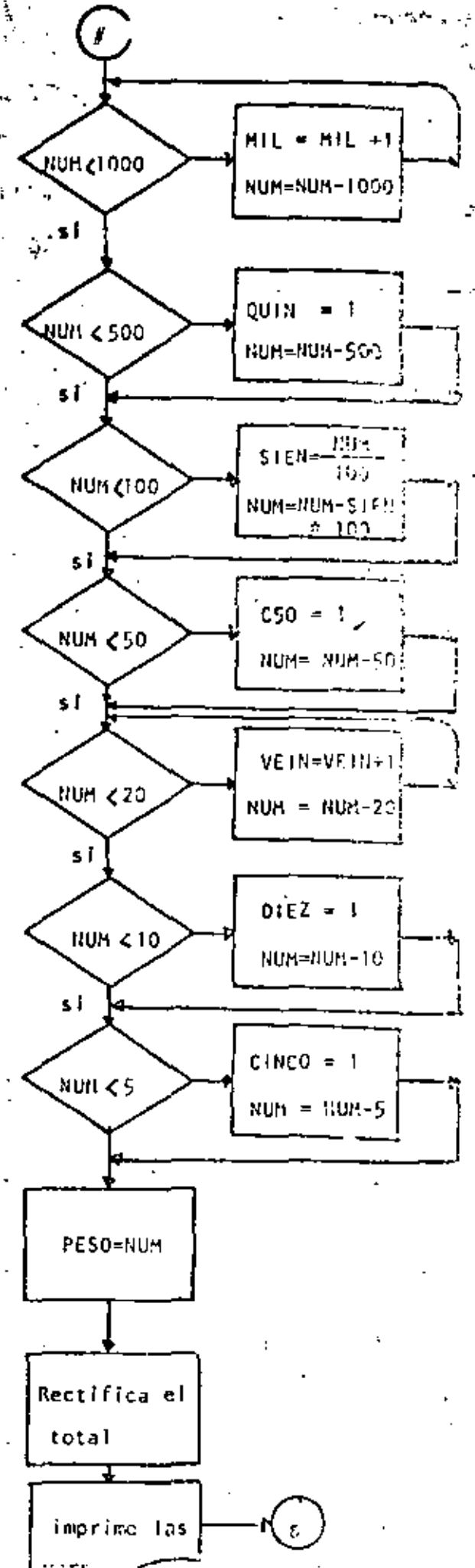
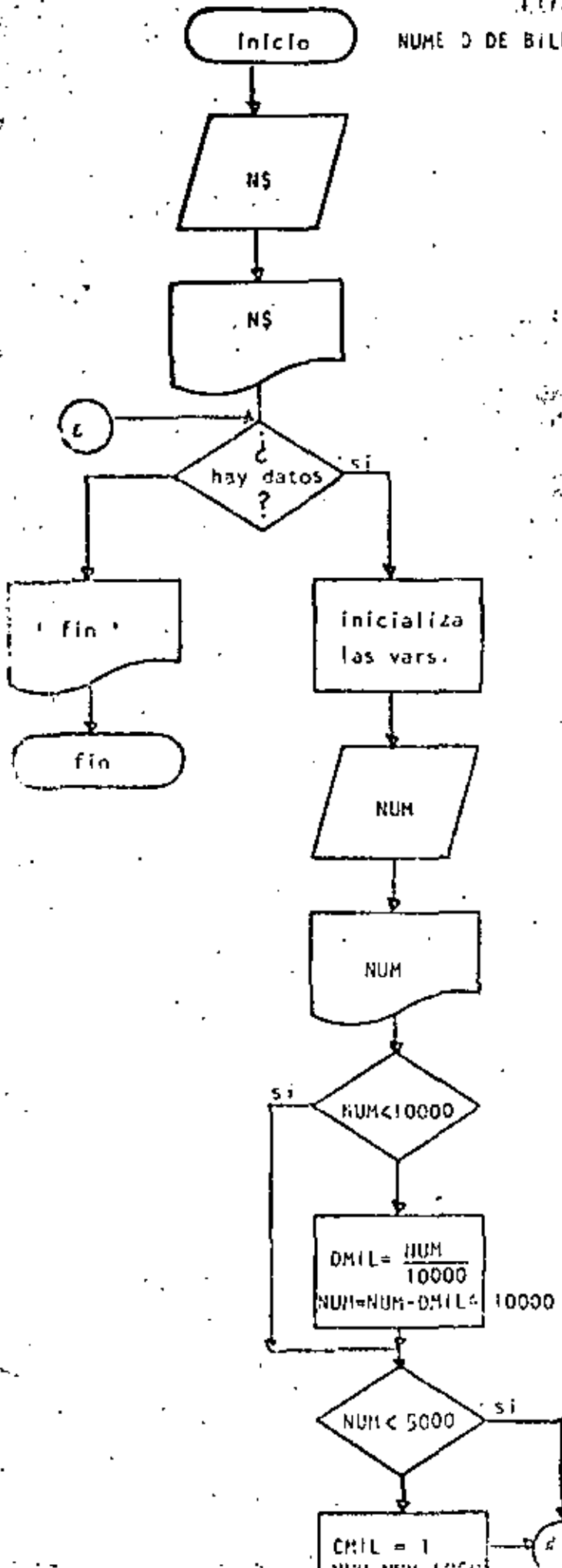

```

10 REM----DOCE----
20 REM CAMBIO DE BASE BINARIA A BASE DECIMAL UTILIZANDO ARREGLOS
30 EN EN BASIC LA DIMENSION POR OMISION ES DE ONCE ( 0 A 10)
40 JM
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 TECLLEAR'
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'
280 GOTO 70
290 'ENDDO
300 PRINT 'FIN DE CAMBIO DE BASE'
310 END
    
```



```

PRUEBA
CAMBIO DE BASES DE DOS A DIEZ
HAY DATOS(SI O NO)? SI
DAME LOS DIEZ DIGITOS BINARIOS
1
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
LIST

1022

```
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 PESO
50 REM
60 INPUT "NOMBRE DEL CLIENTE":IN$
70 PRINT
80 PRINT "HOLA ",IN$, " ESPERO QUE ESTE USTED BIEN"
90 PRINT
100 'DOWHILE HAYA DATOS
110 INPUT "HAY DATOS (SI O NO)":S$ : IF S$ <> "SI" THEN 820
120 CLS:'INICIALIZA VARIABLES
130 REM CIGN 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,C30,VEIN,DIEZ,CINCO,PESOS
160 DATA 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 ' RESTANCIA 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 NUMZ=NUM
290 IF NUMZ < 1000 THEN 330
300 MIL=MIL+1
310 NUMZ=NUMZ-1000
320 GOTO 290
330 ' REVISAR SI ES NECESARIO DAR UN BILLETE DE 500
340 IF NUMZ < 500 THEN 380
350 QUIN=1
360 NUMZ=NUMZ-500
370 ' DETERMINACION DE LOS BILLETES DE 100
380 IF NUMZ < 100 THEN 420
390 SIEN=FIX(NUMZ/100)
400 NUMZ=NUMZ-SIEN*100
410 ' REVISAR SI ES NECESARIO DAR UN BILLETE DE 50
420 IF NUMZ < 50 THEN 440
430 C30=1:NUMZ=NUMZ-50
440 ' DETERMINAR EL NUMERO DE BILLETES ( O MONEDAS ) DE 20
450 IF NUMZ < 20 THEN 480
460 VEIN=VEIN+1
470 NUMZ=NUMZ-20 : GOTO 450
480 ' REVISAR SI ES NECESARIO DAR UNA MONEDA DE DIEZ
490 IF NUMZ < 10 THEN 530
500 DIEZ=1
510 NUMZ=NUMZ-10
520 ' DETERMINAR SI ES NECESARIO DAR UNA MONEDA DE CINCO
530 IF NUMZ < 5 THEN 570
540 CINCO=1
550 NUMZ=NUMZ-5
560 ' LO QUE RESTA ES EL NUMERO DE MONEDAS DE UN PESO
570 PESO=NUMZ
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,TTAL
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; GO TO 100
810 'ENDDO
820 PRINT
830 PRINT "ADIOS " #1
840 END

```

Ready

>RUN
NUMERO 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

>RUH

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.

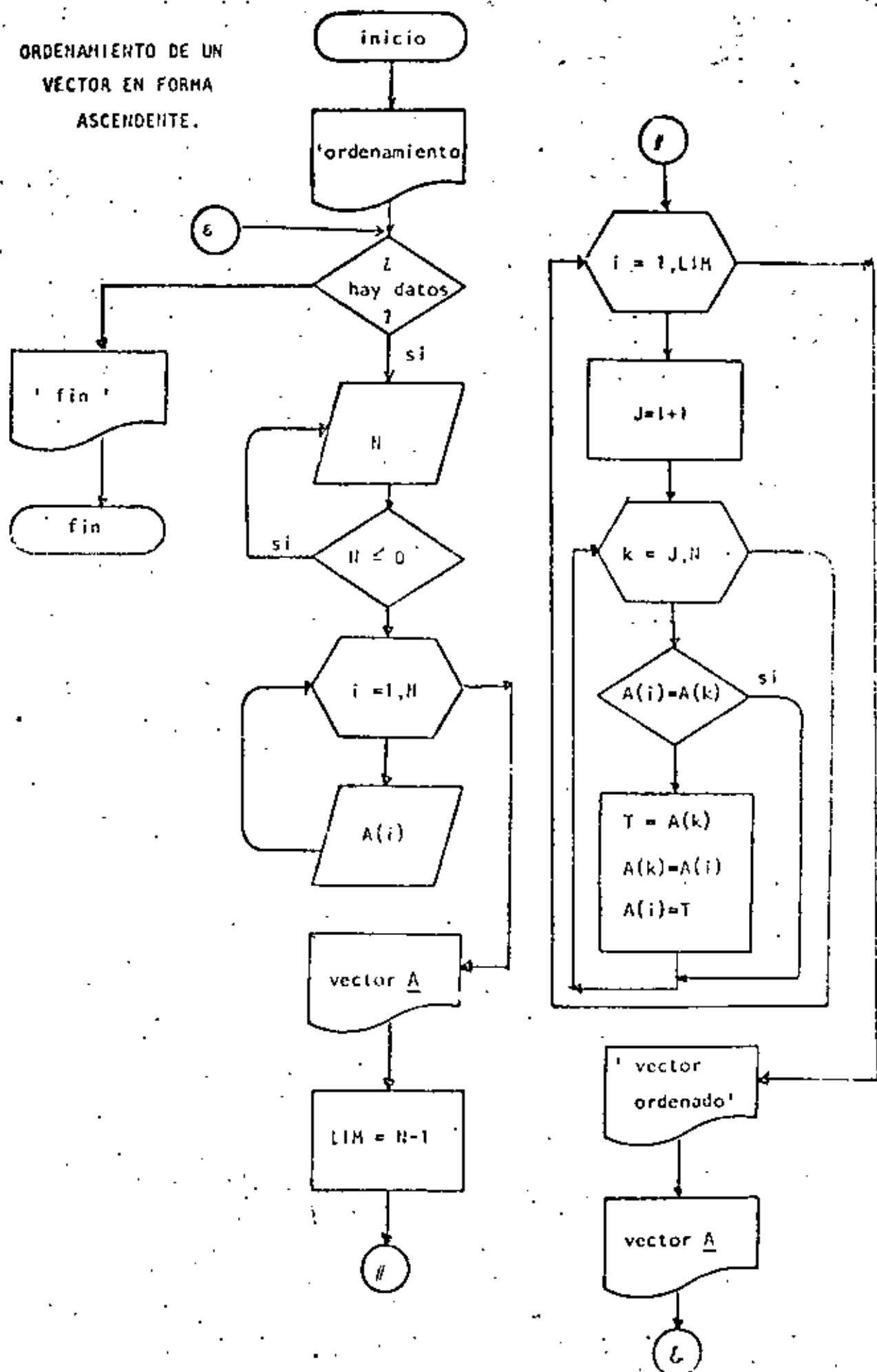
BILLETES DE DIEZ MIL	12	120000
BILLETES DE CINCO MIL	0	120000
BILLETES DE MIL	3	123000
BILLETES DE QUINIENTOS	0	123000
BILLETES DE CIEN	4	123400
BILLETES DE CINCUENTA	1	123450
MONEDAS DE VEINTE	0	123450
MONEDAS DE DIEZ	0	123450
MONEDAS DE CINCO	1	123450
MONEDAS DE UNO	1	123450

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) THEN 240
230 ' A(I) FUE > QUE A(K)
240 T=A(K):A(K)=A(I):A(I)=T
250 ' SE INTERCAMBIARON
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 'END99
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

```

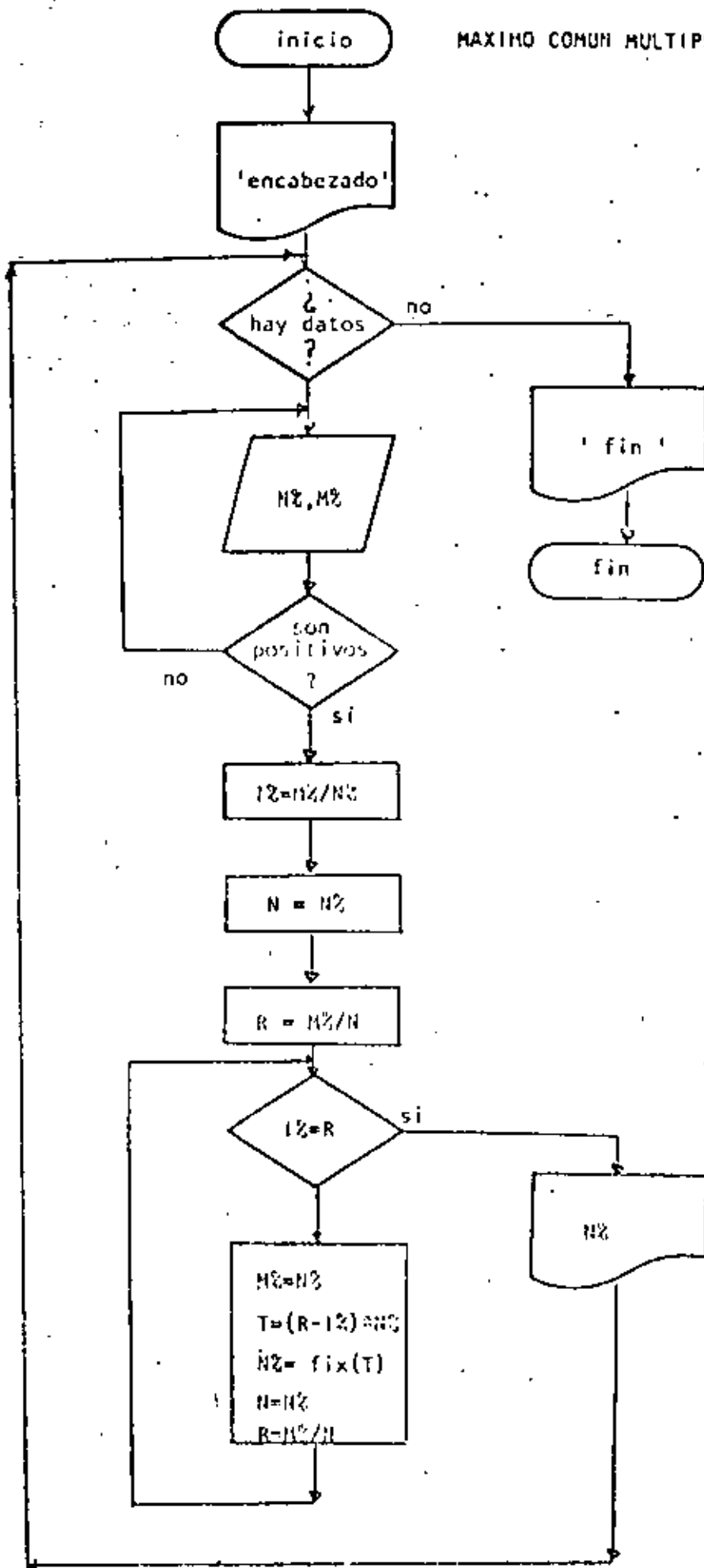
0 3 5 7 9
HAY DATOS (SI O NO)? NO
FIN DEL ORDENAMIENTO

```

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)?:SI:IF SE <> 'SI' THEN 290
90   INPUT "DAME LOS NUMEROS PARA BUSCAR EL M.C.D.:MZ,NZ
100   IF NZ <= 0 OR MZ <= 0 THEN 90
110   ' SE CALCULA EL RESIDUO
120   IZ=MZ/NZ
130   N=MZ
140   R=MZ/N
150   'DOWHILE IZ < R
160   IF IZ = R THEN 240
170   MZ=MZ
180   T=(R-IZ)*NZ
190   NZ=(I)*T
200   IZ=MZ/NZ
210   N=NZ
220   R=MZ/N
230   GOTO 160
240   'ENDDO
250   'N REPRESENTA EL MAXIMO COMUN DIVISOR
260   PRINT "EL MAXIMO COMUN DIVISOR ES : "NZ
270   GOTO 80
280 'ENDDO
290 PRINT " FIN DEL MAXIMO COMUN DIVISOR"

```

SEDE:

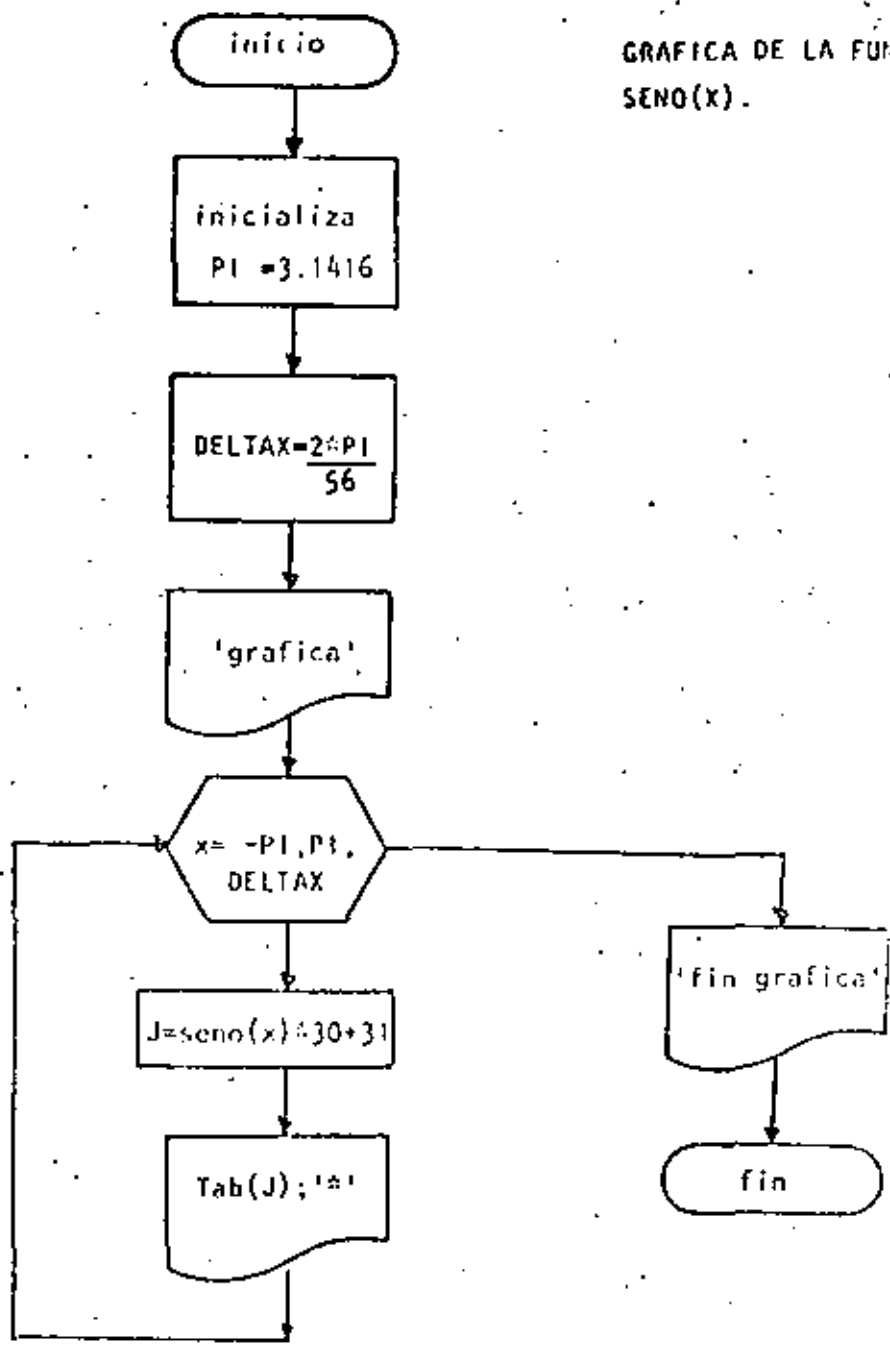
PROGRAMA PARA OBTENER EL MAXIMO COMUN DIVISOR DE DOS NUMEROS
USANDO EL ALGORITMO DE EUCLIDES

```

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)? SI
FIN DEL MAXIMO COMUN DIVISOR

```

GRAFICA DE LA FUNCION
SENO(X).

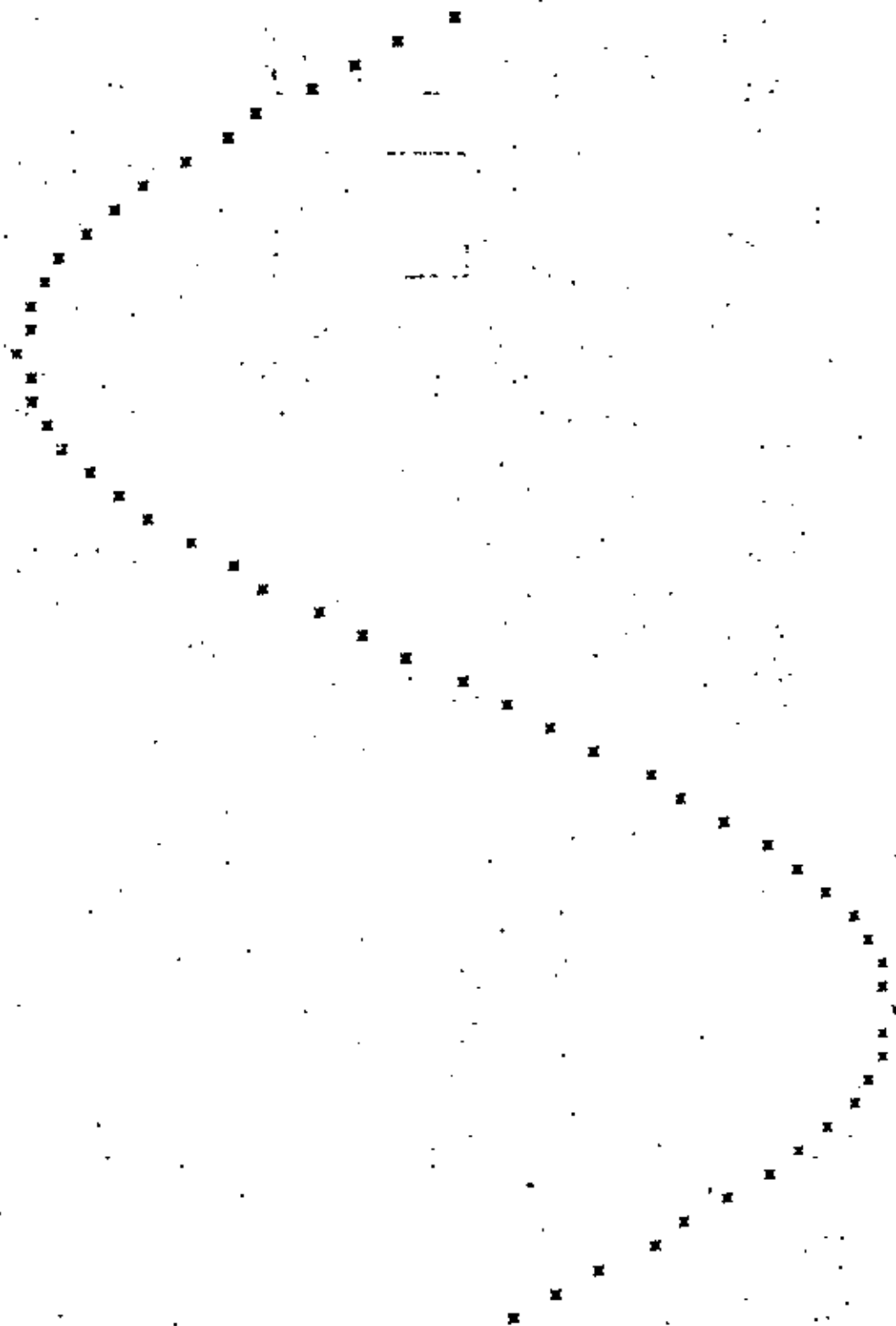


```

10 REM *****DECASEL*****
20 REM GRAFICA DEL SENO
30 REM
40 MOVA 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 PARTICULA ES DE 64 POSICIONES
110     PRINT TAB(6);"X"
120 NEXT X
130 PRINT "FIN DE LA GRAFICA"
140 END
  
```

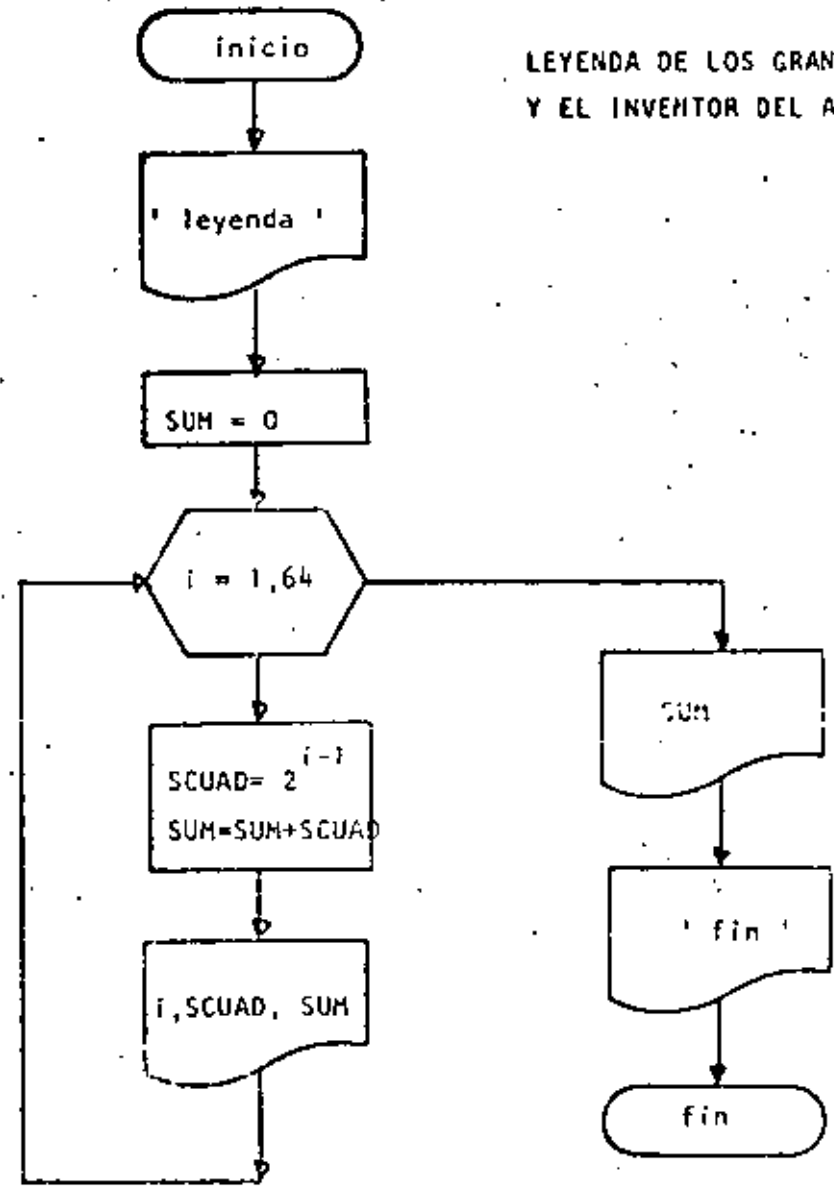
Ready
RUN
GRAFICA DEL SEND

1030



FIN DE LA GRAFICA
Ready

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 CORRO EL INVENTOR
40 REM DEL JUEGO DE AJEDREZ
50 REM
60 PRINT "LEYENDA DE LOS GRANOS DE MAIZ Y EL AJEDREZ":PRINT
70
80 PRINT "CASILLA      GRANOS QUE LE CORRESPONDEN      TOTAL"
90 FOR I=1 TO 64
100   SCUAD=2^(I-1)
110   SUM=SUM+SCUAD
120   PRINT TAB(0);I;TAB(10);SCUAD;TAB(45);SUM
130 PRINT
140 PRINT "TOTAL DE GRANOS":SUM:PRINT
150 PRINT
160 PRINT "FIN DEL GRANERO"
170 END
  
```

Ready
>RUN

LEYENDA DE LOS GRANOS DE MAIZ Y EL AJEDREZ

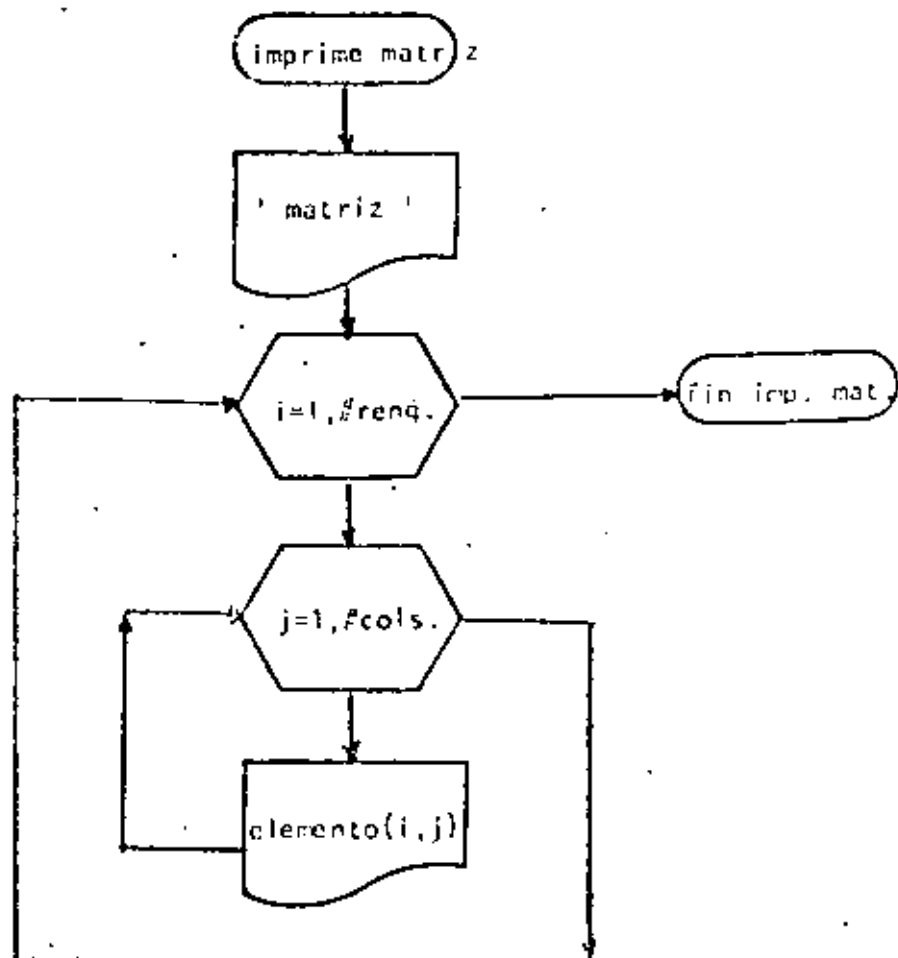
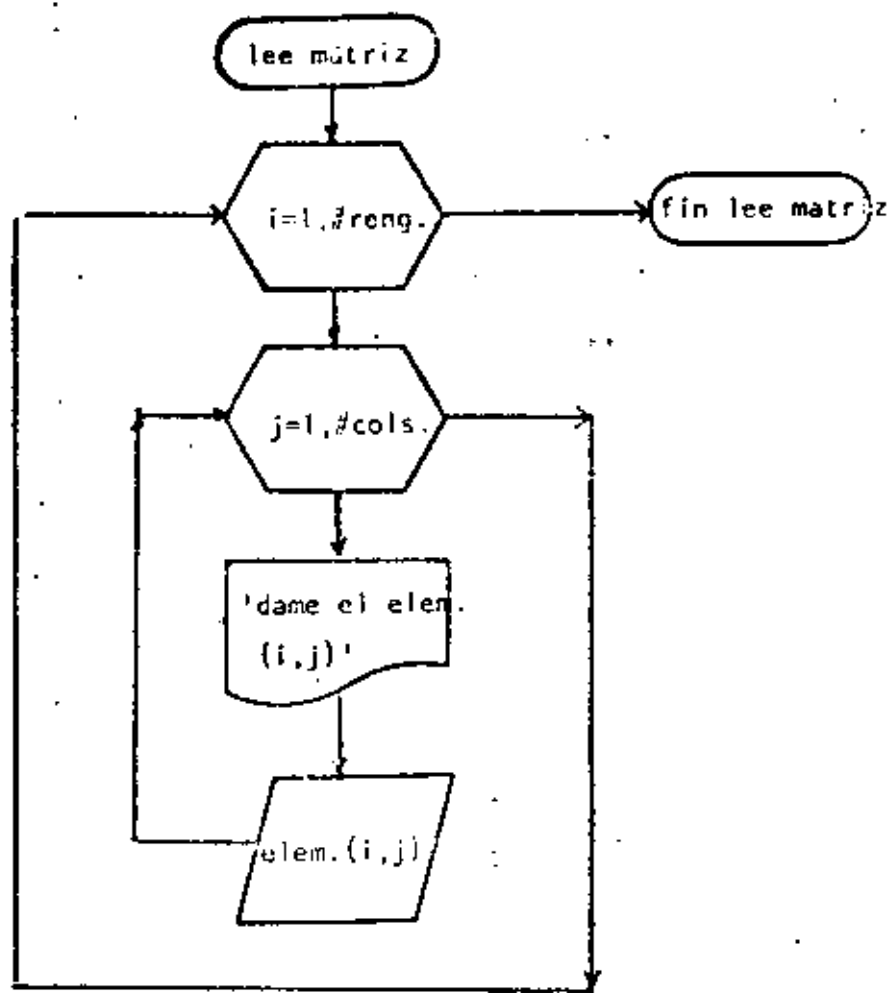
CASILLA	GRANOS QUE LE CORRESPONDEN	TOTAL
1	1	1
2	2	3
3	4	7
4	8	15
5	16	31
6	32	63
7	64	127
8	128	255
9	256	511
10	512	1023
11	1024	2047
12	2048	4095
13	4096	8191
14	8192.005257575	16383.005257575
15	16384	32767.008359375
16	32768.0204375	65535.0204375
17	65536	131071.029196875
18	131072	262143.029196875
19	262144	524287.029196875
20	524288	1048575.029196875
21	1048576	2097151.029196875
22	2097152	4194303.029196875
23	4194304	8388607.029196875
24	8388608	16777215.029196875
25	16777216	33554431.029196875
26	33554432	67108863.029196875
27	67108864	134217815.029196875
28	134217816	268435631.029196875
29	268435632	536871263.029196875
30	536871264	1073742527.029196875
31	1073742528	2147485055.029196875
32	2147485056	4294970111.029196875
33	4294970112	8589940223.029196875
34	8589940224	17179880447.029196875
35	17179880448	34359760895.029196875
36	34359760896	68719521791.029196875
37	68719521792	137439043583.029196875
38	137439043584	274878087167.029196875
39	274878087168	549756174335.029196875
40	549756174336	1099511486671.029196875
41	1099511486672	2199022973343.029196875
42	2199022973344	4398045946687.029196875
43	4398045946688	8796091893375.029196875
44	8796091893376	17592187867551.029196875
45	17592187867552	35184375735103.029196875
46	35184375735104	70368751470207.029196875
47	70368751470208	140737502940415.029196875
48	140737502940416	281475005880831.029196875
49	281475005880832	562950011761663.029196875
50	562950011761664	1125900023523267.029196875
51	1125900023523268	2251800047046535.029196875

52	2251779813685248	4503598150973991
53	4503611438530560	9007209589504551
54	9007199234740792	1.8014388044295590+16
55	1.8014351264091700+16	3.6028760109087270+16
56	3.6028777018963970+16	7.2057557128051260+16
57	7.2057574037727940+16	1.4411515116597920+17
58	1.4411491011873320+17	2.882299612847130+17
59	2.8823037615171170+17	5.7646033748642480+17
60	5.7646075220342350+17	1.1529218897398320+18
61	1.1529245282333230+18	2.3058456120036720+18
62	2.3058439892136940+18	4.6116896272170440+18
63	4.611673227494820+18	9.2233625510248480+18
64	9.2233728060047760+18	1.8446784587871820+19

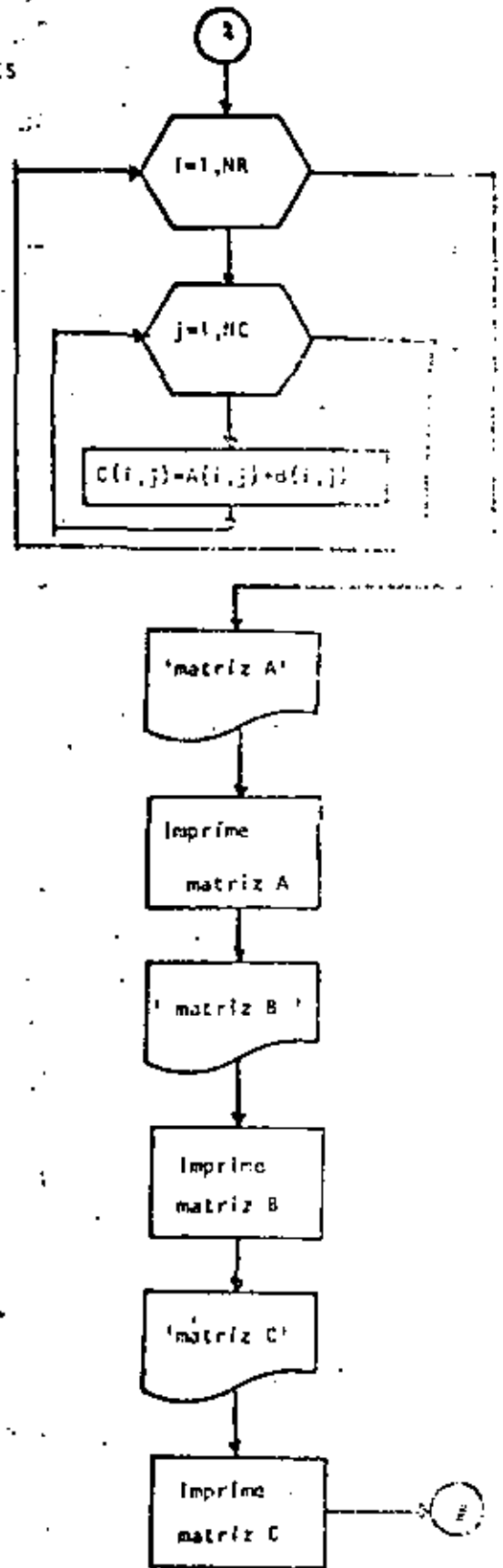
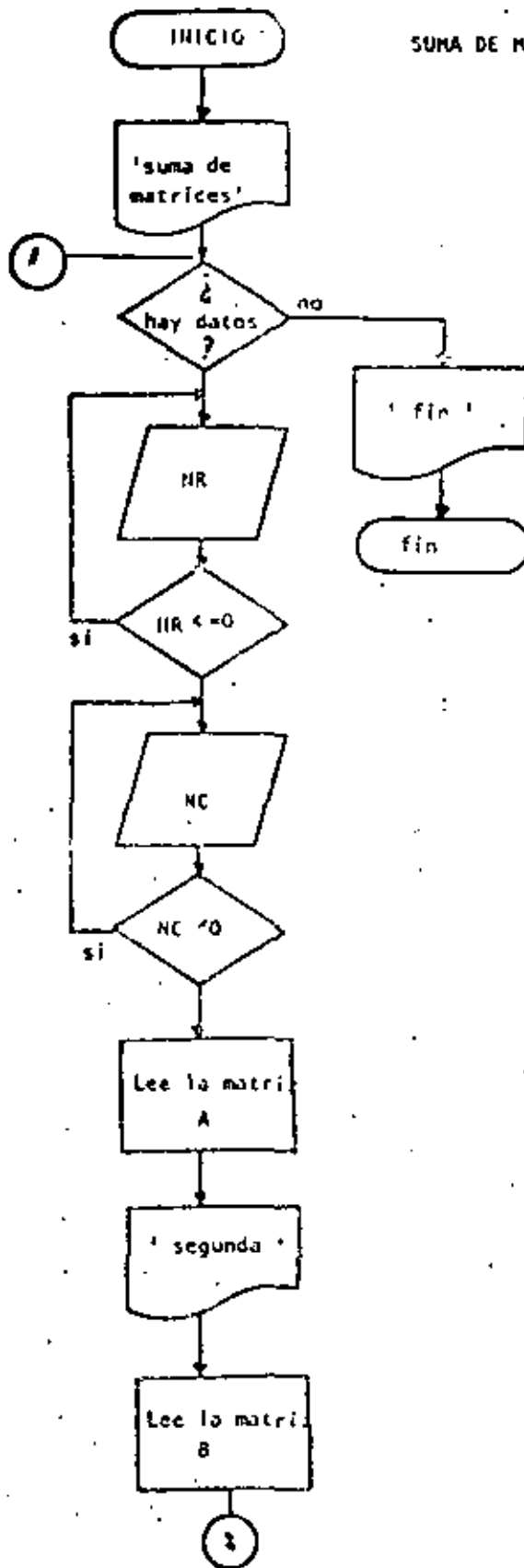
TOTAL DC 056897 1.8446734587871820+19

FIG 021 DR00030

Ready



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 "NO? DATOS(SI O NO)?"$
70 IF $ <> "SI" THEN 830
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 NR
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 "#####.1" 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 "#####.1" 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 "#####.1" C(I,J):
570     NEXT J
580     PRINT
590 NEXT I

```

```
600 PRINT "FIN DE LA SUMA DE MATRICES"
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)? 0
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

1038

>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

$$1 \quad x^2 + 5 \quad x + 3 = 0$$

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

$$1 \quad x^2 + 6 \quad x + 2 = 0$$

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

$$1 \quad x^2 + 0 \quad x + 4 = 0$$

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

$$0 \quad x^2 + 8 \quad x + 9 = 0$$

?/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

$$3 \quad x^2 + 7 \quad x + 5 = 0$$

RAICES COMPLEJAS CONJUGADAS

RAIZ 1 = (-1.16667 , .552771)

RAIZ 2 = (-1.16667 , -.552771)

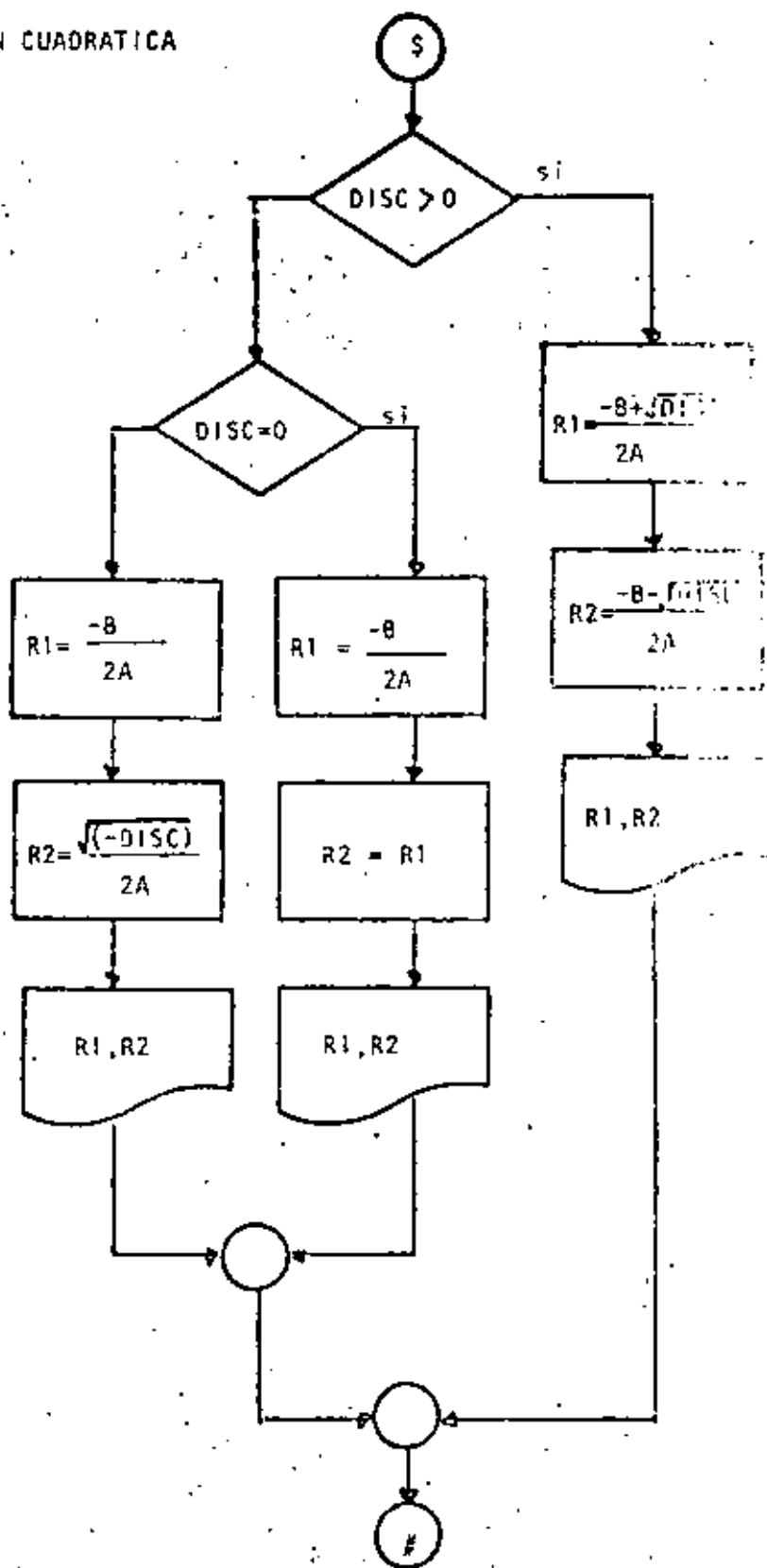
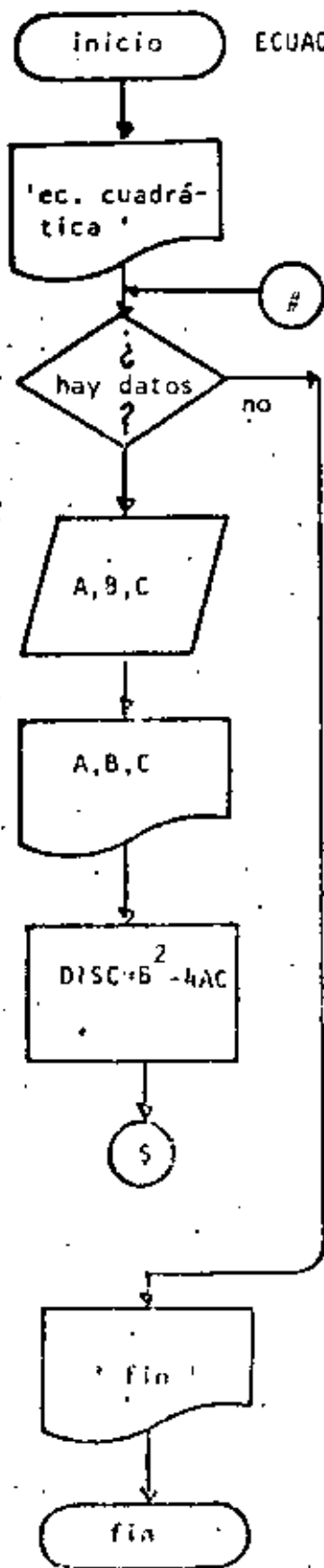
HAY DATOS(SI O NO)? NO

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.

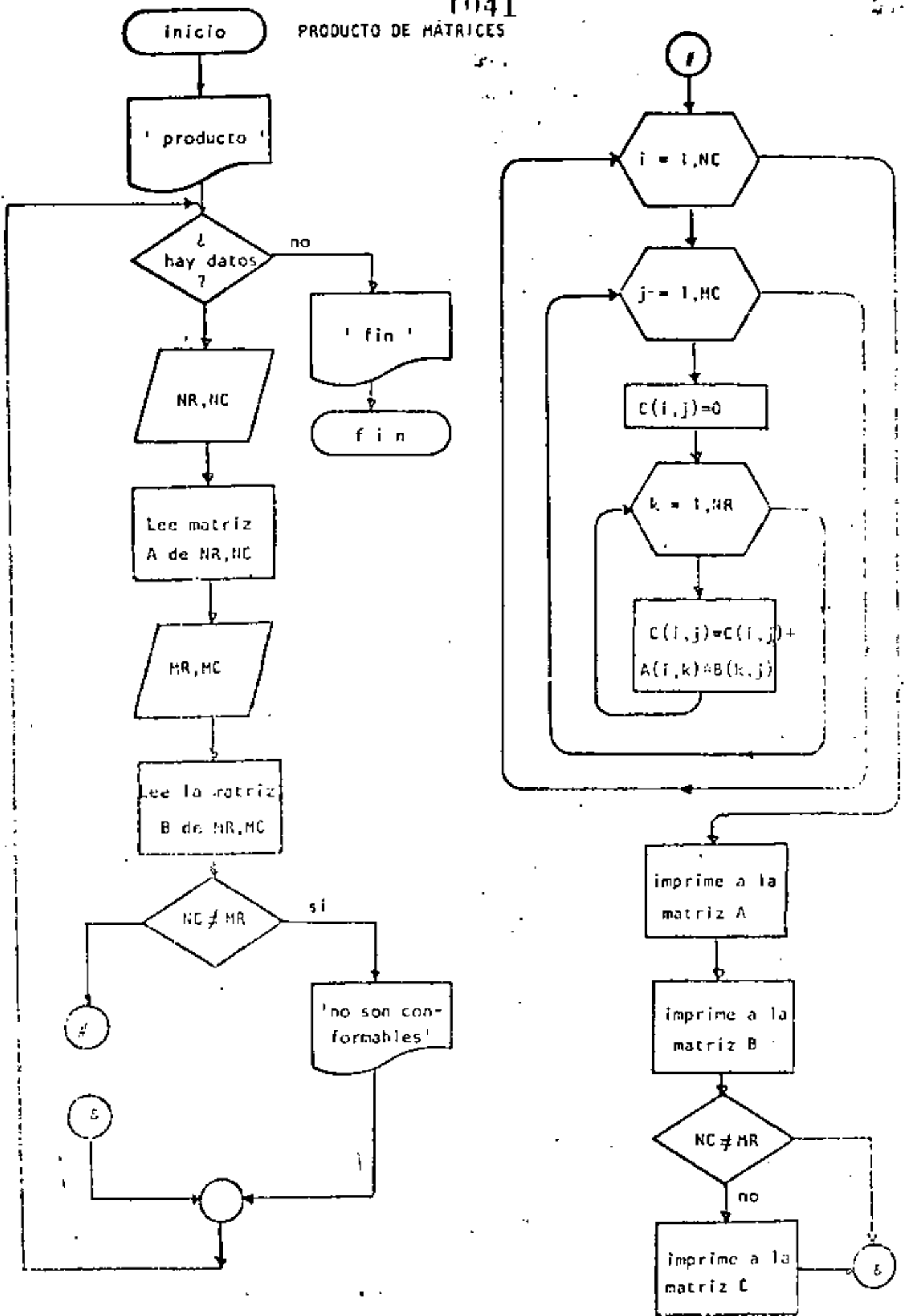
ECUACION CUADRATICA



```

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;"i)"
380 PRINT "RAIZ 2 = ("&R1;"-"&R2;"i)"
390 PRINT
400 'ENDIF
410 'ENDVP
420 'ENDWG
430 DATA 20
440 PRINT "FIN DE ECUACIONES CUADRATICAS"
450 END

```



```

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)":S1
70 IF S1 <> "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 MC <> MR THEN 450: REM NO SON CONFORMABLES
340 FOR I=1 TO MR
350 FOR J=1 TO MC
360 C(I,J)=0
370 FOR K = 1 TO MR
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 GOTO 450: IF MC <> 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 "#####.#####"
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 "#####.#####"
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 'ENDOO
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

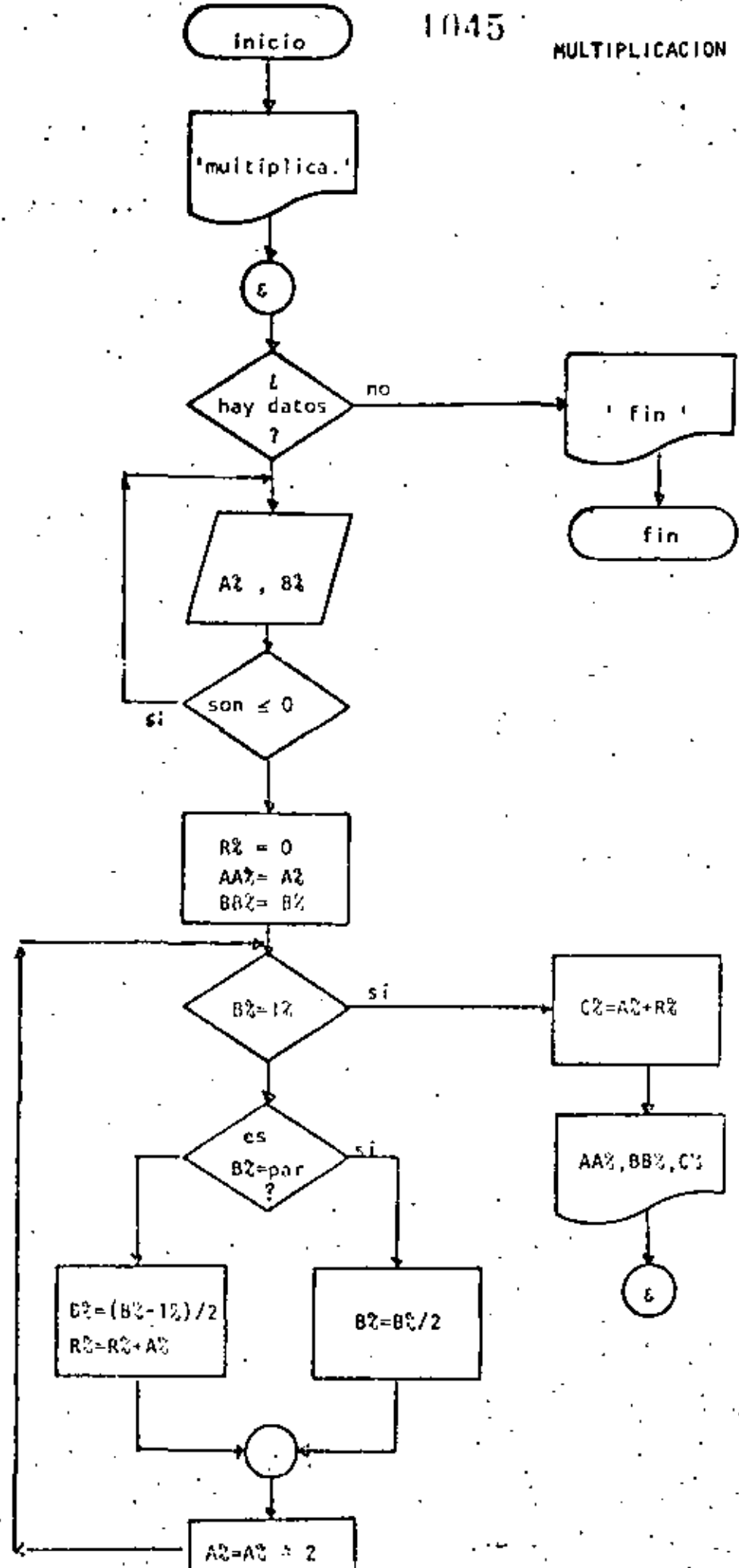
MATRIZ RESULTADO

49.0	96.0	56.0	59.0
39.0	55.0	20.0	16.0

HAY DATOS(SI O NO)? NO

FIN DE LA PRODUCTO DE MATRICES

Ready



```

10 REM-----VEINTIUNO-----
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*:AZ,BZ
100    IF AZ <= 0 OR BZ <= 0 THEN 90
110    RZ=0
120    AAZ=AZ
130    BBZ=BZ
140    *DOWHILE BZ > 1
150    IF BZ = 1Z THEN 200
160        IF BZ = FIX(BZ/2)*2 THEN 180
170            GOTO 210
180        * THEN
190            BZ=BZ/2
200            GOTO 240
210        * ELSE
220            BZ=(BZ-1Z)/2
230            RZ=RZ+AZ
240        *ENDIF
250        AZ=AZ*2
260    GOTO 150
270    *ENDDO
280    CZ=AAZ+BBZ
290    PRINT *A VALE *;AAZ,*B VALE *;BBZ,*EL PRODUCTO ES *;RZ
300    GOTO 70
310 *ENDDO
320 PRINT
330 PRINT* FIN DE LA MULTIPLICACION*
340 END

```

Ready

SRUN

MULTIPLICACION DE 2 NUMEROS UTILIZANDO PRODUCTO Y
DIVISION ENTRE DOS

```

HAY DATOS (SI O NO)? SI
DAME LOS VALORES DE A Y B? 3,9
A VALE 3      B VALE 9      EL PRODUCTO ES 27
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 0,4    B VALE 5      EL PRODUCTO ES 2
HAY DATOS (SI O NO)? NO

```

FIN DE LA MULTIPLICACION

Ready



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

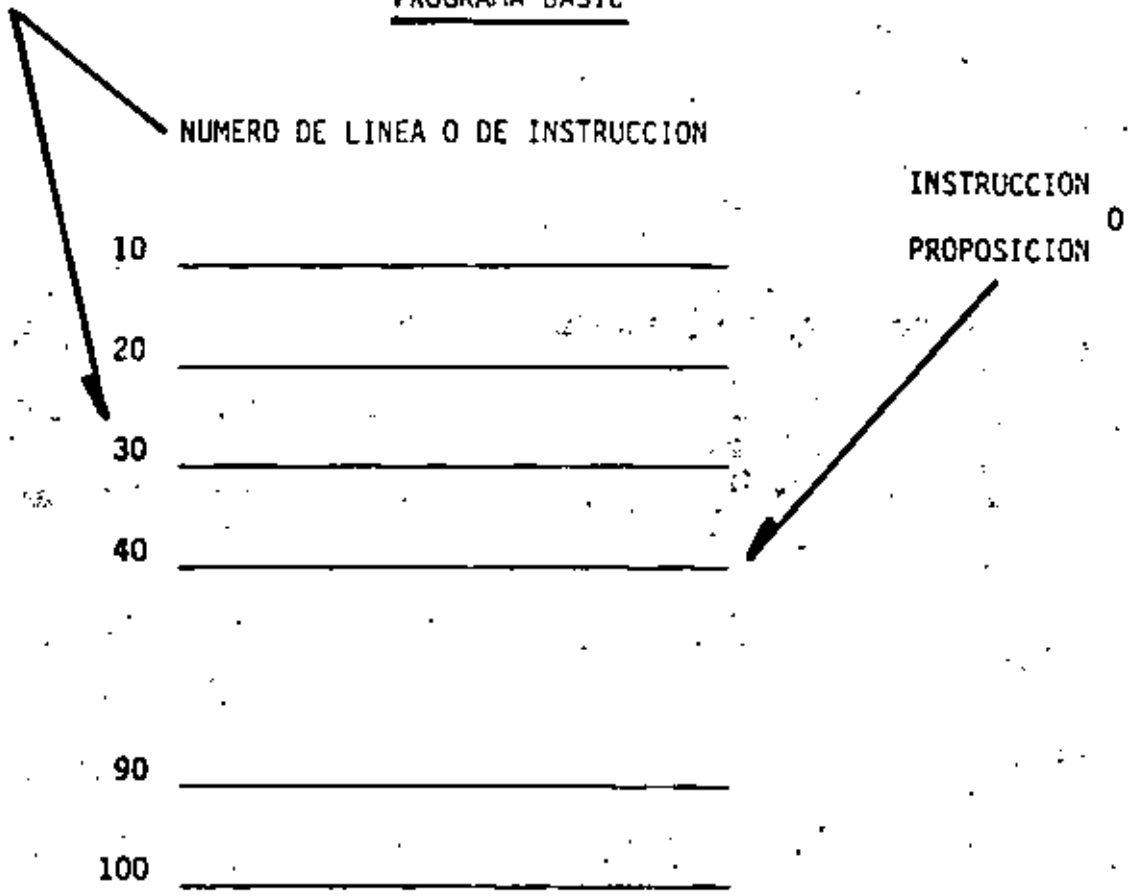
LENGUAJE DE PROGRAMACION BASIC
(PRIMERA PARTE)

COMPLEMENTO
INSTRUCCIONES BASICAS

M. EN C. RICARDO CIRIA MERCE

4/ OCTUBRE, 1983

PROGRAMA BASIC



EJECUCION SECUENCIAL

ASCENDENTE POR NUMERO DE LINEA

EXCEPTO CUANDO ALGUNA INSTRUCCION

INDIQUE OTRA COSA

PROPOSICION

END

10 _____
20 _____
30 _____
40 _____

70 _____
80 END
90 _____
100 _____

- TERMINA LA EJECUCION DEL PROGRAMA
- ES LA ULTIMA INSTRUCCION QUE SE EJECUTA, AUNQUE NO NECESARIAMENTE LA ULTIMA INSTRUCCION DEL PROGRAMA

PROPOSICION

REM

10 _____

20 _____

30 REM PRIEMR PROGRAMA

40 REM CURSO BASIC

50 _____

80 END

90 REM LA LINEA ANTERIOR ES LA

100 REM ULTIMA PROPOSICION QUE SE EJECUTA

110 _____

120 _____

- PERMITE HACER COMENTARIOS SOBRE
EL PROPIO PROGRAMA
- CONSUME MEMORIA

PROPOSICION

LET

PROPOSICION DE ASIGNACION

PERMITE ASIGNAR VALORES A LAS VARIABLES

10 LET A = 2

ASIGNARA EL VALOR DE 2 A LA VARIABLE A

20 LET B = A + 5.36

ASIGNARA EL VALOR 7.36 A LA VARIABLE ~~A~~ B

30 LET A = A + 1

ASIGNARA EL VALOR 3 A LA VARIABLE A

EN GENERAL :

No. LINEA LET VARIABLE = EXPRESION ARITMETICA

EXPRESION ARITMETICA

DESPUES DE SU EVALUACION, OBTENDREMOS

SIEMPRE UN VALOR NUMERICO (NUMERO)

10 LET A = 1

20 LET B = 2

30 LET B = 3

LA E.A. MAS SIMPLE ES UNA CONSTANTE

SE PUEDEN COMBINAR CONSTANTES Y VARIABLES
CON LOS SIGUIENTES OPERADORES, PARA FOR-
MAR E.A. MAS COMPLEJAS:

+ ADICION

- SUBSTRACCION

* MULTIPLICACION

/ DIVISION

↑ EXPONENCIACION ([])

A + B

A + B + C

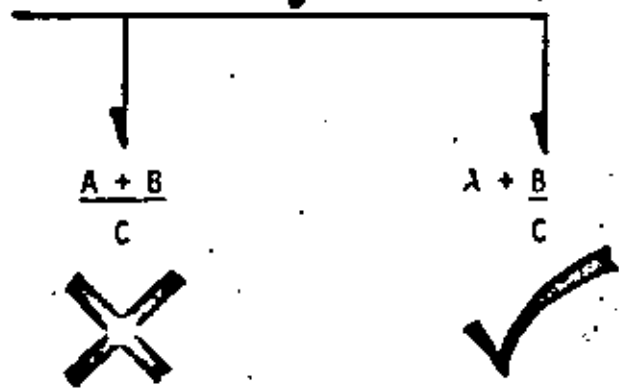
B * C

C / B

EXPRESION ARITMETICA (CONT.)

$$A + B / C$$

?



CONVENCION

— PRIORIDADES :

- 1 ↑
- 2 * , /
- 3 + , -

— ORDEN DE EVALUACION



IZQUIERDA A DERECHA

EXPRESION ARITMETICA (CONT.)

$$\frac{A + B}{C}$$



$$(A + B) / C$$

LOS PARENTESIS MODIFICAN LAS

PRIORIDADES DE LOS OPERADORES:

SE EVALUARA PRIMERO AQUELLA (S)

PARTE (S) DE LA E.A. QUE SE.

ENCUENTRE (N) (ENTRE PARENTESIS)

```

100 REM *** EXPRESION  ARITMETICA ***
110 A = 1
120 B = 2
130 C = 3
140 D = 4
150 E = 5
160 REM
170 X1 = A + D / B
180 X2 = (A + D) / B
190 X3 = (A + B) / C + D
200 X4 = A + B - C
210 X5 = A + B * C
220 X6 = A * B ! C
230 X7 = (A * B) ! C
240 X8 = A ! B + C
250 X9 = A ! (B + C)
260 REM
270 PRINT A , B , C , D , E
280 PRINT
290 PRINT X1 , X2 , X3 , X4 , X5
300 PRINT
310 PRINT X6 , X7 , X8 , X9
320 END

```

1	2	3	4	5
3	2.5	5	0	7
8	8	4	1	

```

100 S1$ = "          CURSO BASIC"
110 S2$ = "EXPRESION ARI"
120 S3$ = "METICA"
130 A = 2.35
140 B = 3.708
150 C = (A + B * (A / B) / (A - B)) + 1
160 PRINT S1$
170 PRINT S2$ ; S3$
180 PRINT "C = (A + B * (A / B) / (A - B)) + 1 = " , C
190 PRINT S2$;
200 PRINT S3$
210 END

```

```

          CURSO BASIC
EXPRESION ARITMETICA
C = (A + B * (A / B) / (A - B)) + 1 = 1.6195139912
EXPRESION ARITMETICA

```



PERMITE DESPLEGAR EN PANTALLA
VALORES DE VARIABLES,
CONSTANTES
NUMERICAS O
ALFANUMERICAS

```
10 A = 1  
20 B = 2  
30 A $ = "HOLA"  
40 PRINT A, B, A$  
50 END
```

1

2

HOLA

No LINEA

PRINT

LISTA DE VARIABLES

PROPOSICION

PRINT

(continúa)

LISTA DE VARIABLES

- NULA PRINT
- UNA VARIABLE PRINT V1
- MAS DE UNA VARIABLE
SEPARADAS POR:
 - COMAS : SE DESPLIEGA EN POSICIONES
PREESTABLECIDAS EN LA
PANTALLA
 - PUNTOS Y COMAS: SE DESPLIEGAN LAS VARIABLES
EN FORMA CONTINUA.

POSICIONES PRE ESTABLECIDAS:

1

16

32

48

EJEMPLOS

```

10 A = 150
20 AS = "ABC"

30 PRINT A
40 PRINT
50 PRINT A, A, AS
60 PRINT "EL VALOR DE A ES :" ;A
70 PRINT AS ; "DEF" , AS
80 PRINT A ; A ;
90 END
    
```

	1	16	32
30	150		
40			
50	150	150	ABC
60	EL VALOR DE A ES: 150		
70	ABCDEF	ABC	
80	150	150	

NO HAY ESPACIO , POR SER STRING

UN ESPACIO, POR SER NUMERICO

PROPOSICION

INPUT

PERMITIR CAPTAR O LEER VALORES

DE VARIABLES,

NUMERICAS O

ALFANUMERICAS

10 A = 3

20 A\$ = "HOLA"

30 INPUT A

40 INPUT A \$

50 PRINT A, A \$

60 END

EN UNA PROPOSICION INPUT, ES POSIBLE CAPTAR EL VALOR
DE UNA O MAS VARIABLES.

EN GENERAL:

NO. LINEA INPUT LISTA DE VARIABLES

PROPOSICION INPUT (Continúa)

PARA EL EJEMPLO ANTERIOR :

30 INPUT A, A \$

(TECLEADO)

? 3.76, HOLA

PARA PROPORCIONAR DATOS ALFANUMERICOS

QUE CONTENGAN:

- BLANCOS (AL PRINCIPIO O AL FINAL)
- COMAS (,) &
- DOS PUNTOS (:)

HABRA QUE ENCERRARLOS ENTRE COMILLAS

(TECLEADO)

? 3.76, " HOLA : ALOH "

PROPOSICION

INPUT

(Continúa)

```
10 INPUT      A, AS
20 PRINT      A, AS
30 END
```

> RUN

? 7 + 3

? REDO

? 10

?? "ESTO ES UNA COMA , "

10 ESTO ES UNA COMA ,

APARECERA ?? CUANDO SE HAN PROPORCIONADO MENOS DATOS DE
LOS INDICADOS POR EL INPUT.

PROPOSICION INPUT (Continúa)

ES POSIBLE DESPLEGAR UN MENSAJE EN LA PANTALLA
PARA SOLICITAR UN INPUT

10 INPUT "DAME TU NOMBRE" ; NO \$
EQUIVALENTE A :

10 PRINT " DAME TU NOMBRE " ;

20 INPUT NO \$

? REDO

APARECERA ESTE MENSAJE CUANDO LOS DATOS TECLEADOS
NO COINCIDAN CON EL TIPO INDICADO EN EL INPUT.

? EXTRA IGNORED

APARECERA CUANDO SE PROPORCIONEN MAS DATOS DE LOS
INDICADOS EN EL INPUT

```

10 REM PROGRAMA PARA SUMAR N NUMEROS.
20 A = 1
30 C = 1
40 PRINT "LA SUMA DEL 1 AL " ; C ; " ES : " ; A
50 C = C + 1
60 A = A + C
70 GO TO 40
80 END

```

```

LA SUMA DEL 1 AL 1 ES : 1
LA SUMA DEL 1 AL 2 ES : 3
LA SUMA DEL 1 AL 3 ES : 6
LA SUMA DEL 1 AL 4 ES : 10
LA SUMA DEL 1 AL 5 ES : 15
LA SUMA DEL 1 AL 6 ES : 21
LA SUMA DEL 1 AL 7 ES : 28
LA SUMA DEL 1 AL 8 ES : 36
LA SUMA DEL 1 AL 9 ES : 45
LA SUMA DEL 1 AL 10 ES : 55
LA SUMA DEL 1 AL 11 ES : 66
LA SUMA DEL 1 AL 12 ES : 78
LA SUMA DEL 1 AL 13 ES : 91
LA SUMA DEL 1 AL 14 ES : 105
LA SUMA DEL 1 AL 15 ES : 120
LA SUMA DEL 1 AL 16 ES : 136
LA SUMA DEL 1 AL 17 ES : 153
LA SUMA DEL 1 AL 18 ES : 171
LA SUMA DEL 1 AL 19 ES : 190
LA SUMA DEL 1 AL 20 ES : 210
LA SUMA DEL 1 AL 21 ES : 231
LA SUMA DEL 1 AL 22 ES : 253
LA SUMA DEL 1 AL 23 ES : 276
LA SUMA DEL 1 AL 24 ES : 300
LA SUMA DEL 1 AL 25 ES : 325
LA SUMA DEL 1 AL 26 ES : 351
LA SUMA DEL 1 AL 27 ES : 378
LA SUMA DEL 1 AL 28 ES : 406
LA SUMA DEL 1 AL 29 ES : 435
LA SUMA DEL 1 AL 30 ES : 465
LA SUMA DEL 1 AL 31 ES : 496
LA SUMA DEL 1 AL 32 ES : 528
LA SUMA DEL 1 AL 33 ES : 561
LA SUMA DEL 1 AL 34 ES : 595
LA SUMA DEL 1 AL 35 ES : 630
LA SUMA DEL 1 AL 36 ES : 666
LA SUMA DEL 1 AL 37 ES : 703

```

(BREAK)

```

100 REM PROGRAMA PARA OBTENER LOS N PRIMEROS ELEMENTOS DE
110 REM LA SERIE DE FIBONACCI.
120 C = 1
130 F1 = 1
140 PRINT "EL ELEMENTO ";C;" DE LA SERIE DE FIBONACCI ES "; F1
150 C = 2
160 F2 = 1
170 PRINT "EL ELEMENTO ";C;" DE LA SERIE DE FIBONACCI ES "; F2
180 C=3
190 F = F1 + F2
200 PRINT "EL ELEMENTO ";C;" DE LA SERIE DE FIBONACCI ES "; F
210 F1 = F2
220 F2 = F
230 C = C + 1
240 GO TO 190
250 END

```

```

EL ELEMENTO 1 DE LA SERIE DE FIBONACCI ES 1
EL ELEMENTO 2 DE LA SERIE DE FIBONACCI ES 1
EL ELEMENTO 3 DE LA SERIE DE FIBONACCI ES 2
EL ELEMENTO 4 DE LA SERIE DE FIBONACCI ES 3
EL ELEMENTO 5 DE LA SERIE DE FIBONACCI ES 5
EL ELEMENTO 6 DE LA SERIE DE FIBONACCI ES 8
EL ELEMENTO 7 DE LA SERIE DE FIBONACCI ES 13
EL ELEMENTO 8 DE LA SERIE DE FIBONACCI ES 21
EL ELEMENTO 9 DE LA SERIE DE FIBONACCI ES 34
EL ELEMENTO 10 DE LA SERIE DE FIBONACCI ES 55
EL ELEMENTO 11 DE LA SERIE DE FIBONACCI ES 89
EL ELEMENTO 12 DE LA SERIE DE FIBONACCI ES 144
EL ELEMENTO 13 DE LA SERIE DE FIBONACCI ES 233
EL ELEMENTO 14 DE LA SERIE DE FIBONACCI ES 377
EL ELEMENTO 15 DE LA SERIE DE FIBONACCI ES 610
EL ELEMENTO 16 DE LA SERIE DE FIBONACCI ES 987
EL ELEMENTO 17 DE LA SERIE DE FIBONACCI ES 1597
EL ELEMENTO 18 DE LA SERIE DE FIBONACCI ES 2584
EL ELEMENTO 19 DE LA SERIE DE FIBONACCI ES 4181 (BREAK)

```



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

LENGUAJE DE PROGRAMACION BASICO

INTRODUCCION A LA PROGRAMACION ESTRUCTURADA

NOVIEMBRE, 1983

LENGUAJE DE PROGRAMACION BASIC

INTRODUCCION A LA PROGRAMACION ESTRUCTURADA

SEPTIEMBRE, 1983

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, 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

```

10 'DESGLOSE DE CANTIDADES
20 INPUT CP
25 LPRINTCP
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

```

1000

0

0 1 4 1 2 0 1 5

1000

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)
- 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.

```

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 ELSE LPRINT "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<= V1 THEN 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 GOTO 80 ELSE CP=CP-V3:N3=N3+U:GOTO 70
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:GOTO 100
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
 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 nuevas 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

PROGRAMA
ESTRUCTURADO

<pre> 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 label 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 </pre>	<pre> ① IF p THEN A function B function ② IF q THEN ③ IF t THEN G function ④ DOWHILE u H function ④ ENDDO I function ③ (ELSE) ③ ENDIF ② ELSE C function ③ DOWHILE r D function ③ ENDDO ③ IF s THEN F function ③ ELSE E function ③ ENDIF ② ENDIF ② IF v THEN J function ② (ELSE) ② ENDIF ① ELSE ② IF w THEN M function ② ELSE L function ② ENDIF ① ENDIF K function END function </pre>
--	--

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 GOTO 90
40 IF W GOTO 70
50 GOSUB 390
60 GOTO 200
70 GOSUB 400
80 GOTO 200
90 IF Q GOTO 240
100 GOSUB 410
120 GOSUB 430
130 IF NOT R GOTO 160
140 GOSUB 440
150 GOTO 130
160 IF S GOTO 220
170 GOSUB 450
180 IF NOT V GOTO 200
190 GOSUB 460
200 GOSUB 470
210 END
220 GOSUB 480
230 GOTO 180
240 IF T GOTO 280
250 GOSUB 410
260 GOSUB 420
270 GOTO 340
280 GOSUB 410
290 GOSUB 420
300 GOSUB 490
310 IF NOT U GOTO 340
320 GOSUB 500
330 GOTO 310
340 IF NOT T GOTO 360
350 GOSUB 510
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:

10

IF P THEN

GOSUB 1000:

GOSUB 1010:

GOSUB 1015:

11

ELSE

IF W THEN

GOSUB 1180:

ELSE

GOSUB 1190:

ENDIF

20 GOSUB 1200 :END

1000 PRINT "A":RETURN

1010 PRINT "B":RETURN

1015

IF Q THEN

GOSUB 1020

ELSE

GOSUB 1090

1016 RETURN

1020

IF T THEN

GOSUB 1040:

GOSUB 1050:

GOSUB 1080:

1030 RETURN

1040 PRINT "G":RETURN

1050

IF U THEN

GOSUB 1070:

GOTO 1050:

1060 RETURN

1070 PRINT "H":U=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

11

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

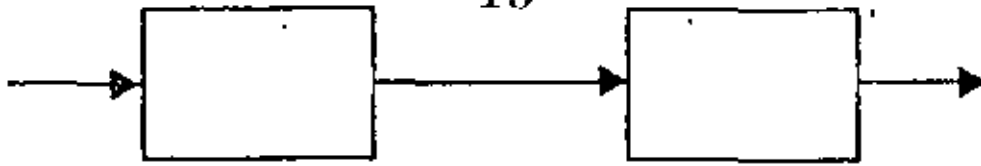
(1) Bohm, C. and Jacopini, G. "Flow Diagrams, Turing Machines and Languages With only two formation Rules" Com. A.C.M. 9, No. 3 (Mayo 1966), 366-371.

SALIDA

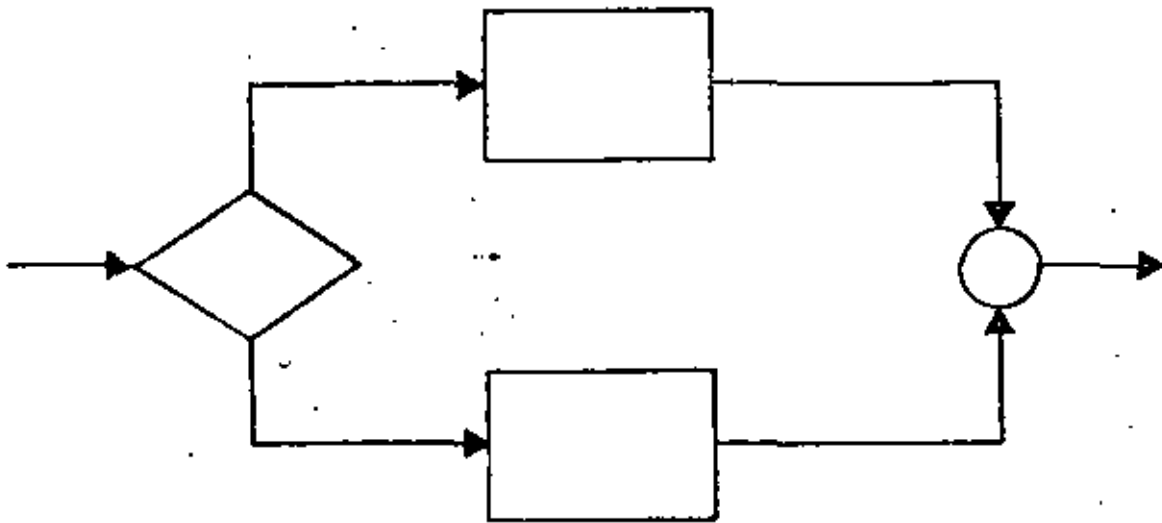
Un programa propio o correcto. (Una entrada y una salida).

ESTRUCTURAS LÓGICAS BÁSICAS

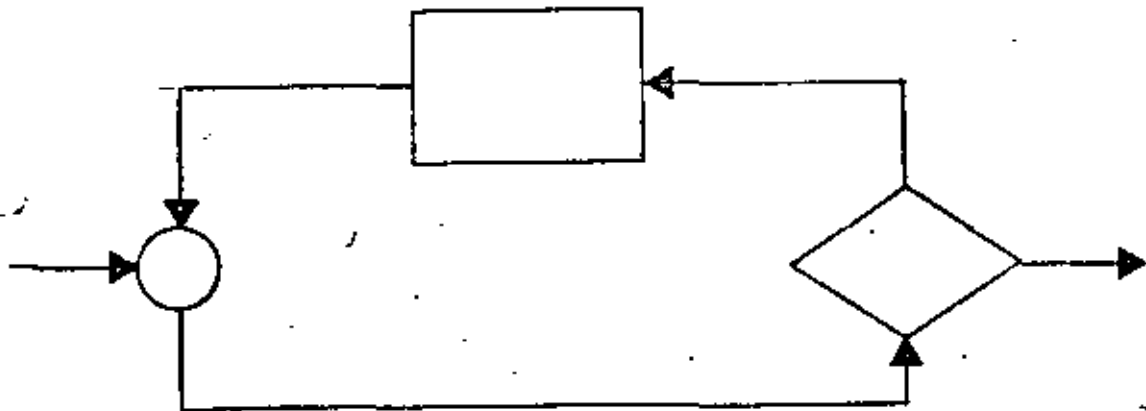
13



SEQUENCE



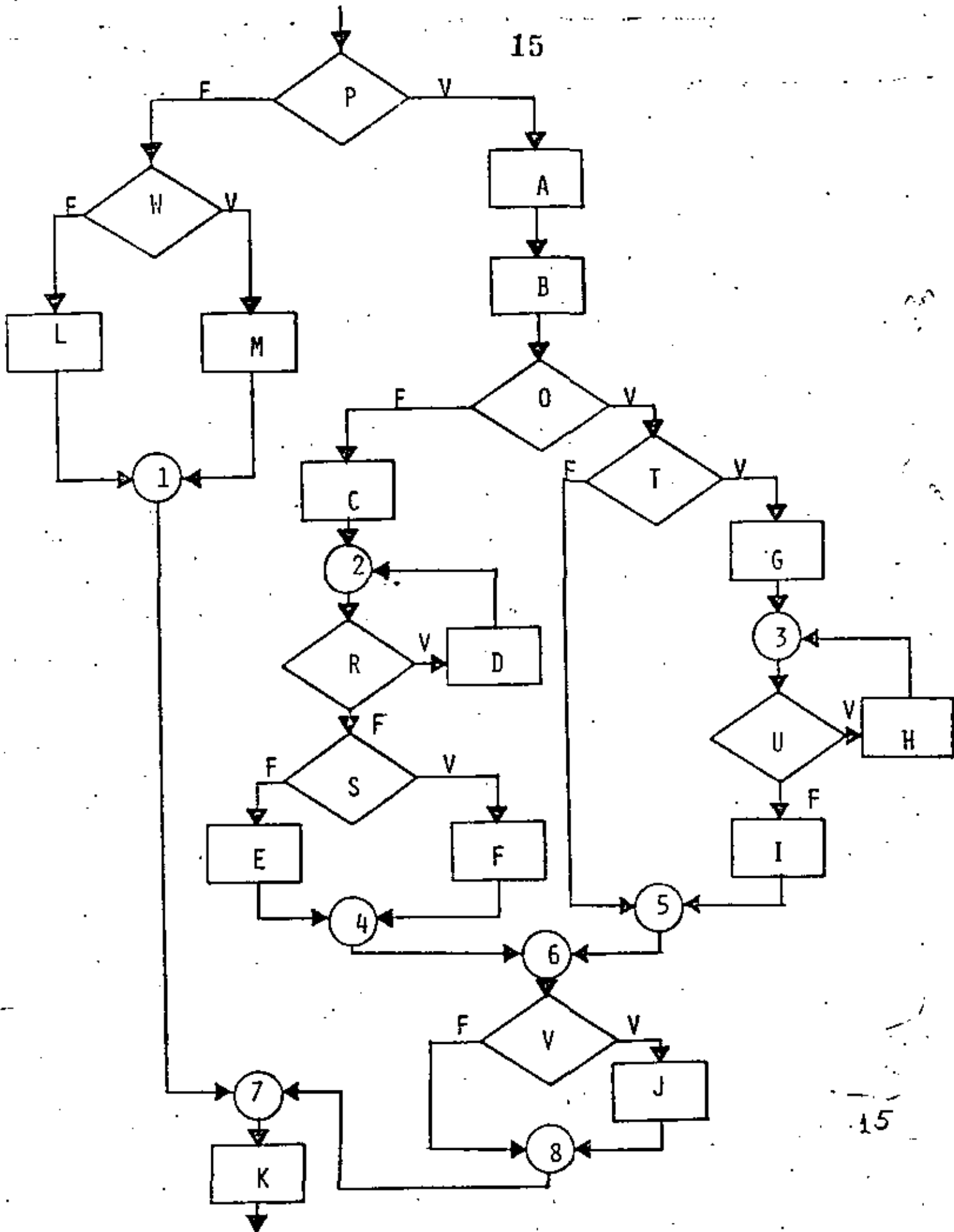
IF THEN ELSE

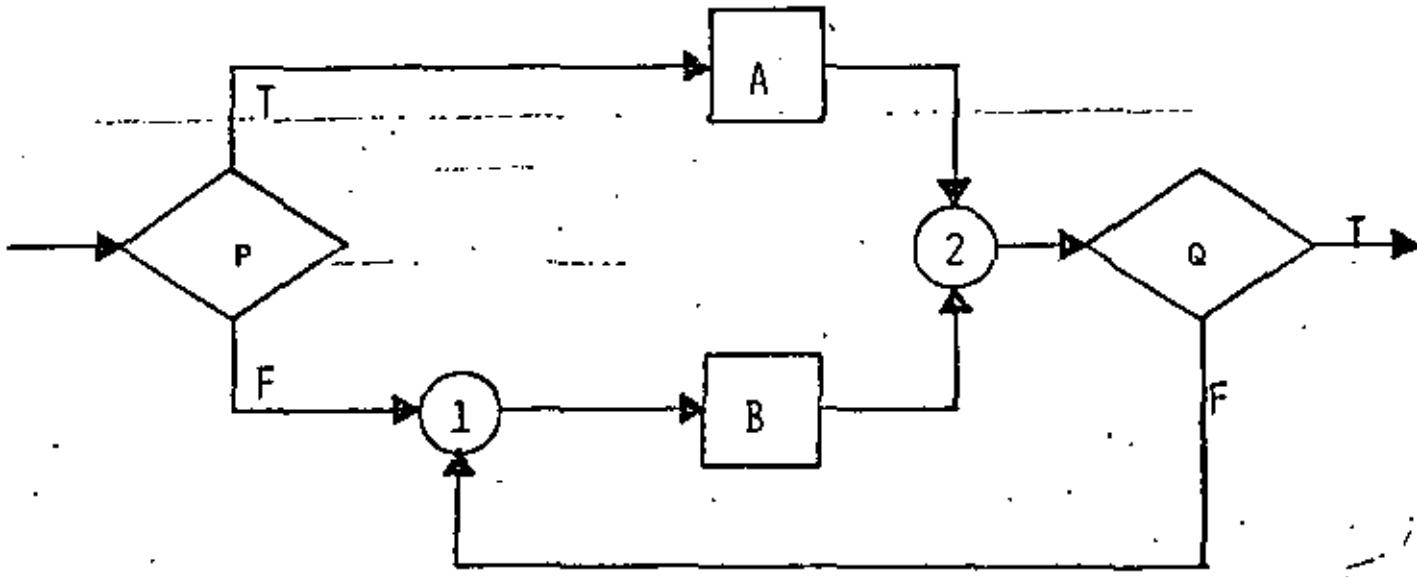
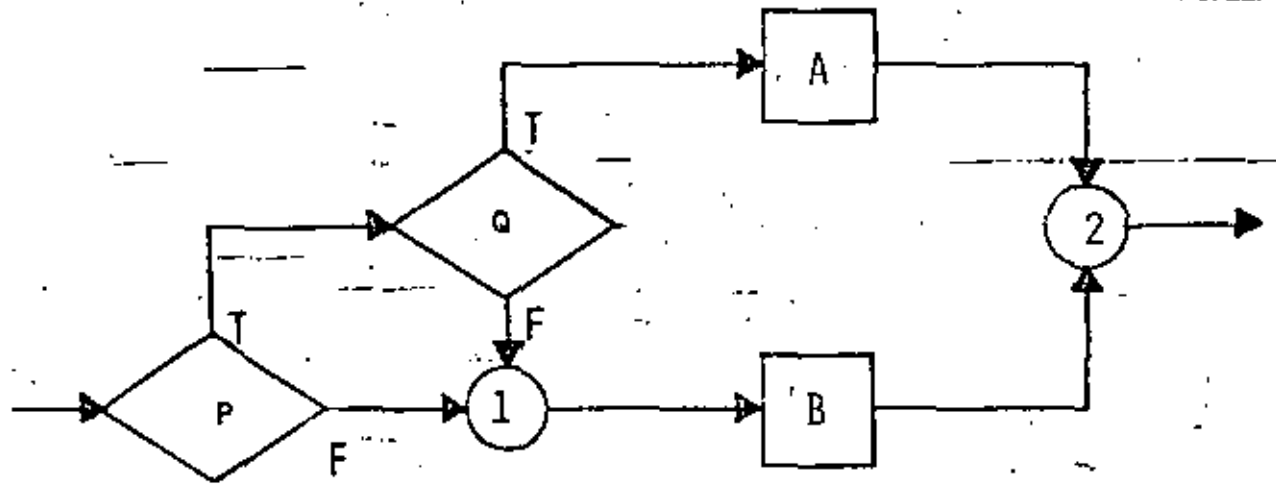
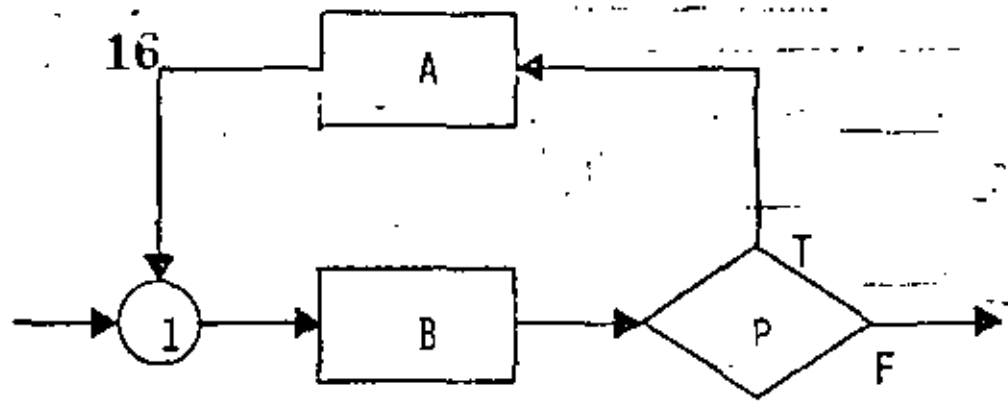


DO WHILE

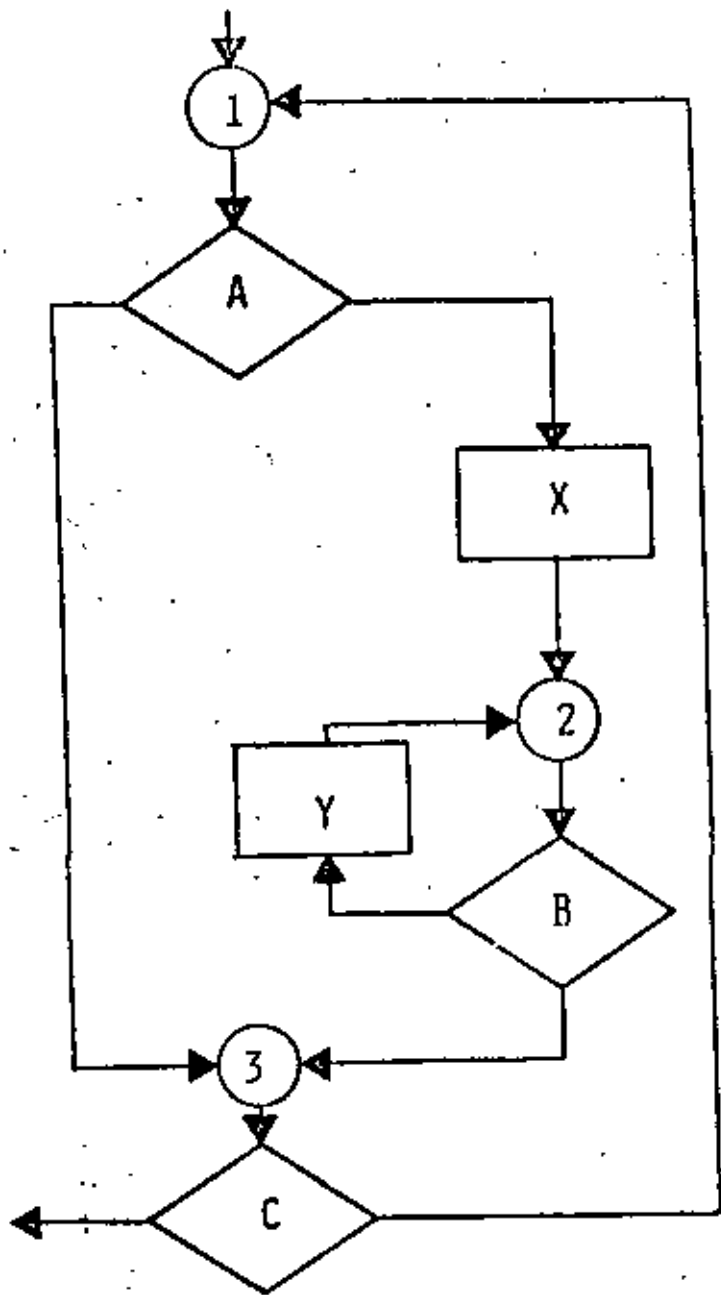
CARACTERISTICAS DE UN PROGRAMA PROPIO

- 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

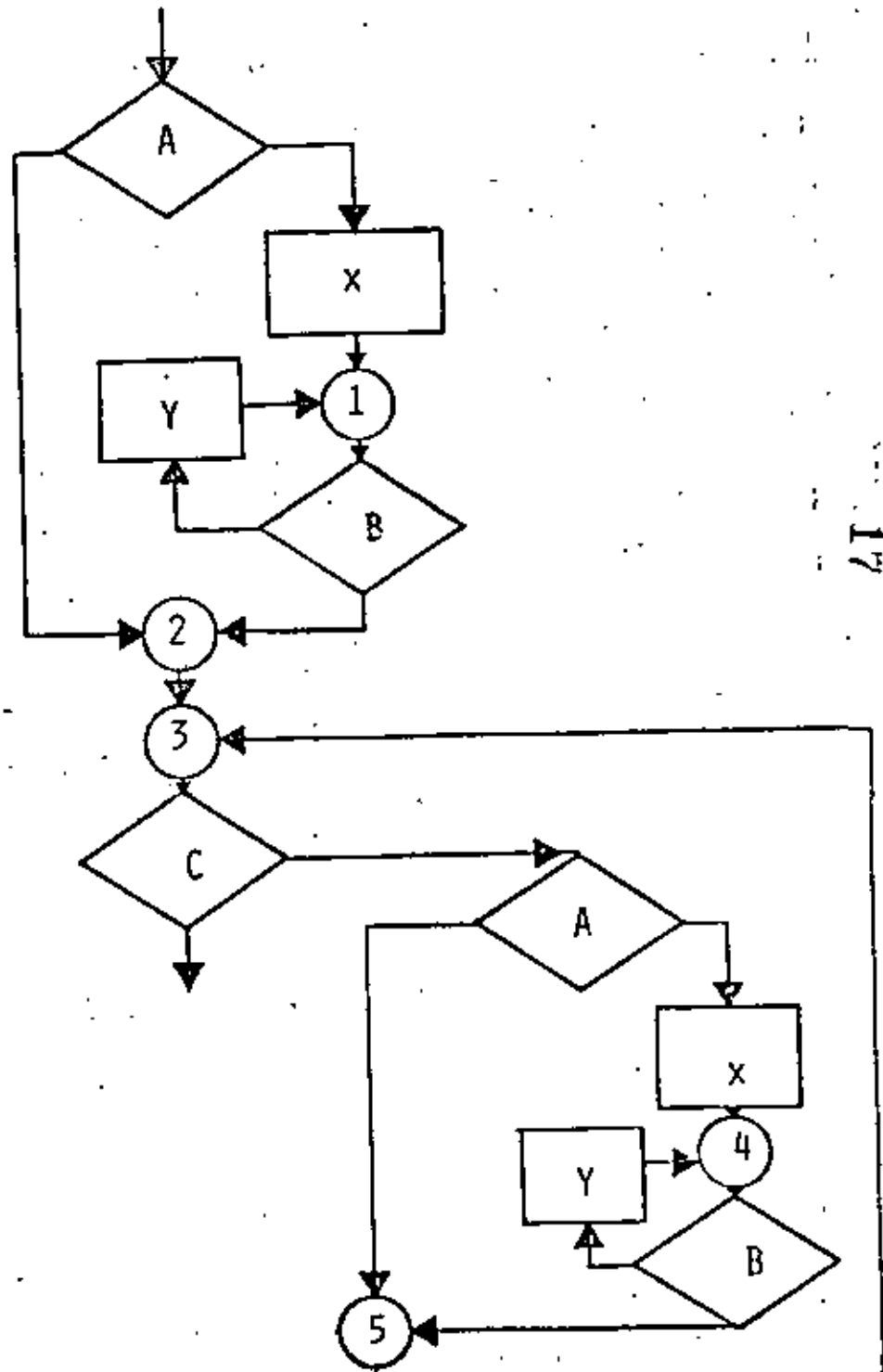




PROBLEMA



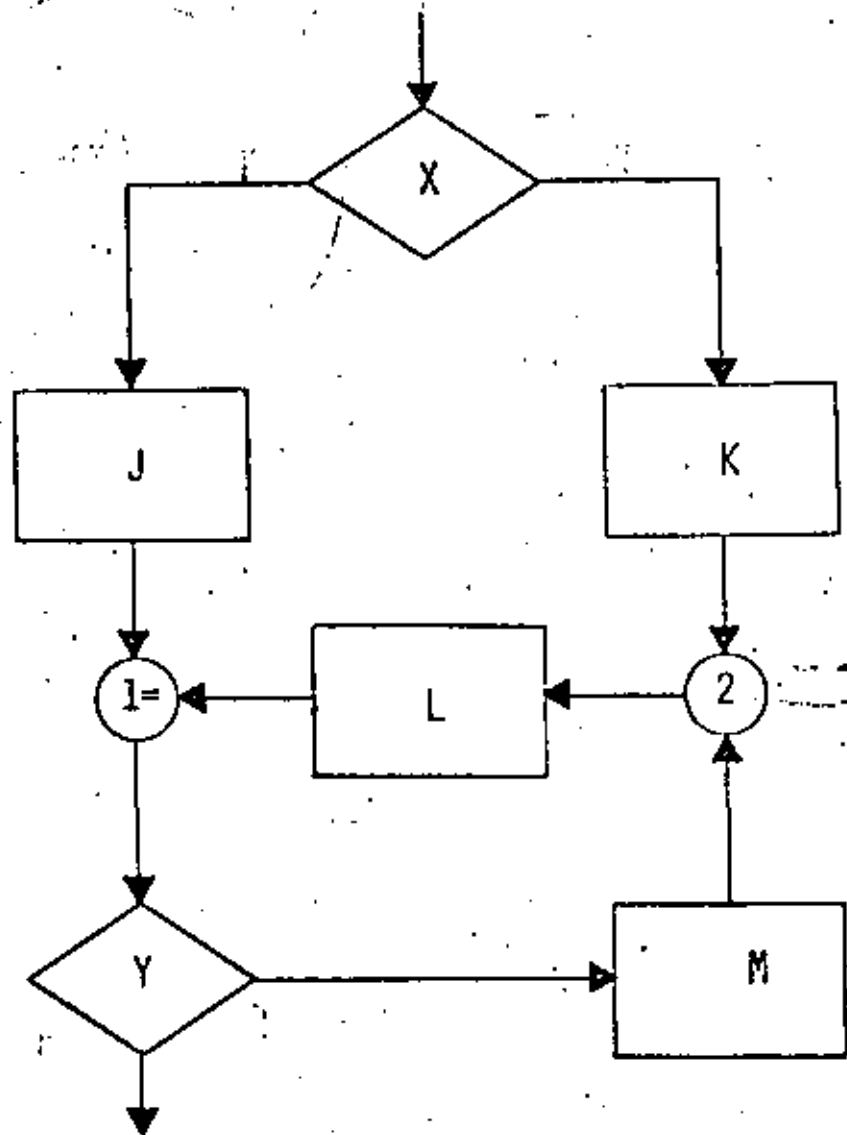
SOLUCION



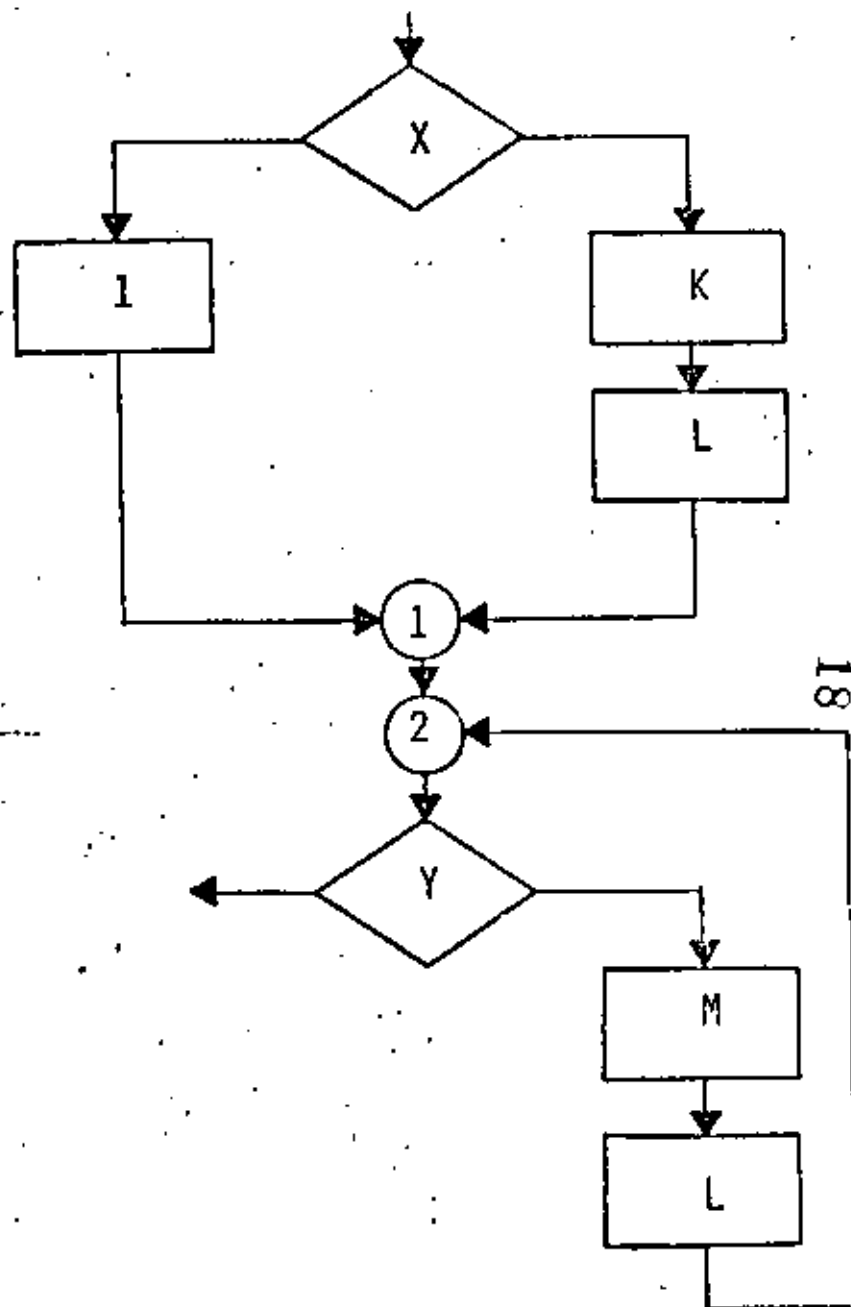
17

17

PROBLEMA



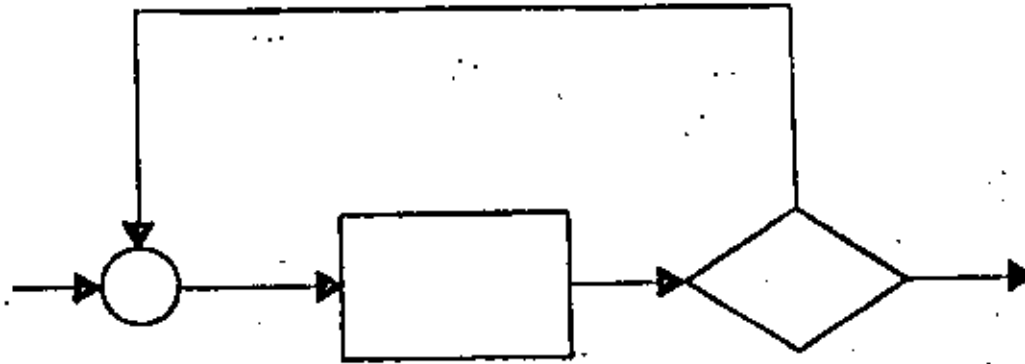
SOLUCION



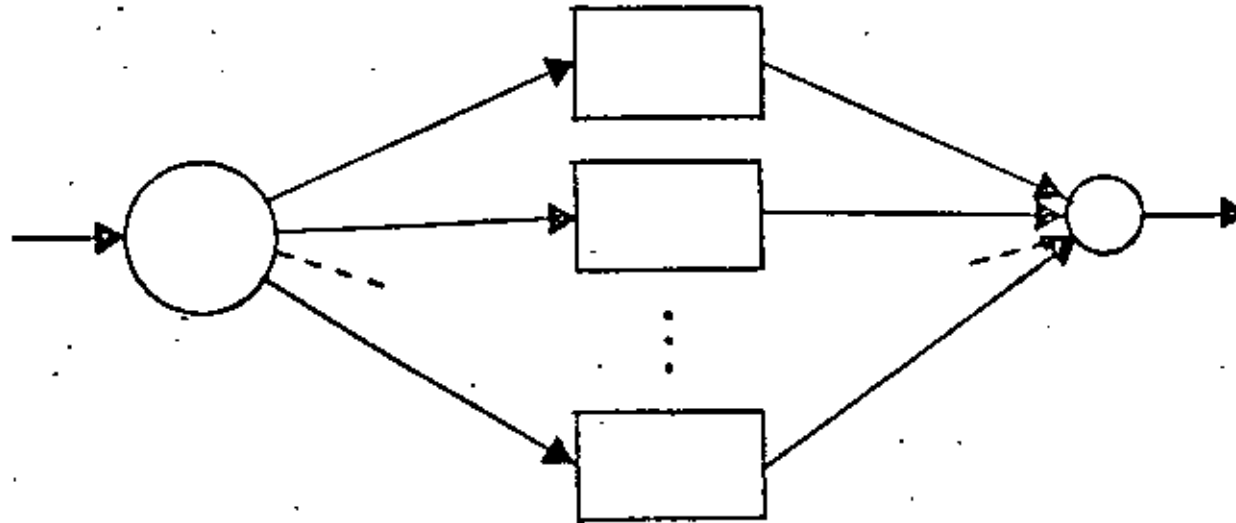
18

18

FIGURAS LOGICAS ADICIONALES
(UNA ENTRADA Y UNA SALIDA)



UNTIL



CASE

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 SUBROUTINAS 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).

SISTEMA:

AUTOR:

HOJA DE

SUBSISTEMA:

REFERENCIA:

FECHA:

APLICACION:

FUNCION: CONVENCIONES DE PSEUDO CODIGO

ENTRADA

Cualquier problema de programación.

Programación Estructurada.

PROCESO

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

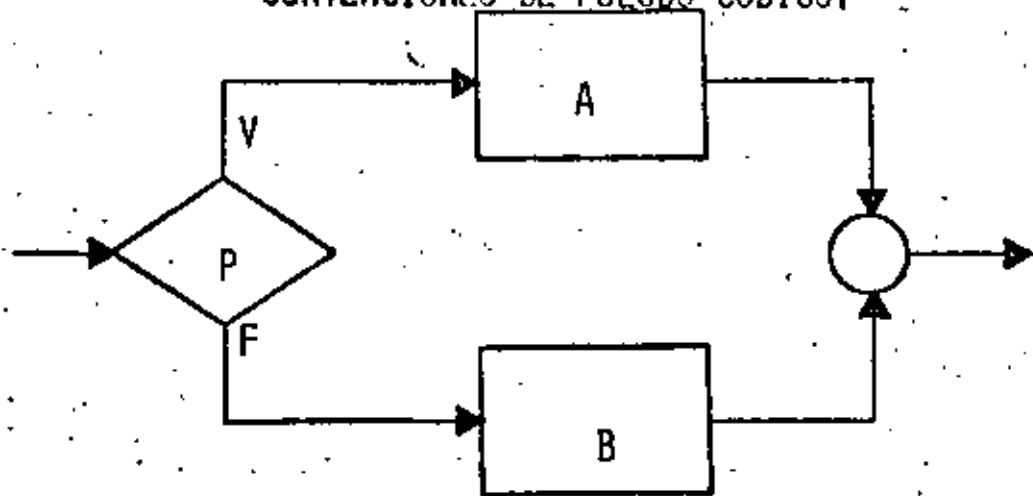
4.- Mediante un corrimiento a la derecha de tres o más posiciones (sangrado) se muestra la dependencia de una estructura respecto a otra.

5.- Todas las operaciones y condiciones se describen usando todas las facilidades existentes en nuestro idioma.

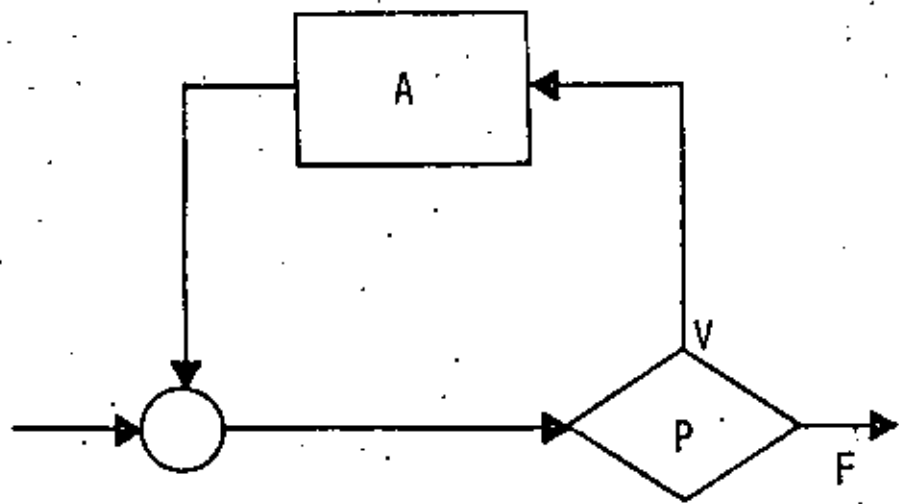
SALIDA

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

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

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

ENDIFENDIF

LEE A,B,C

ENDDO

24

CALCULA DISCRIMINANTE

$$DIS = B^2 - 4 A C$$

RAICES REALES DIFERENTES

25

$$RA\ 1 = (-B + \sqrt{DIS}) / (2A)$$

$$RA\ 2 = (-B - \sqrt{DIS}) / (2A)$$

IMPRIME RA 1

IMPRIME RA 2

RAICES REALES IGUALES

$$RA\ 1 = -B/A$$

$$RA\ 2 = RA\ 1$$

IMPRIME RA 1

IMPRIME RA 2

RAICES COMPLEJAS

$$PARE\ 1 = -B/2A$$

$$PARE\ 2 = PARE\ 1$$

$$PAIM\ 1 = \sqrt{-DIS} / 2A$$

$$PAIM\ 2 = PAIM\ 1$$

IMPRIME PARE 1 " + " PAIM 1 " IMA "

IMPRIME PARE 2 " - " PAIM 2 " IMA "

25

26 Lee VECTOR UNO; (al final poner HV)

Lee VECTOR DOS ; (al final poner HV)

Iniciar i, j

DOWHILE Elemento ; \neq HV δ elementos; \neq HV

DOWHILE Elemento ; > elemento

Solo en 2

ENDDO

DOWHILE Elemento ; > elemento ;

Solo en 1

ENDDO

DOWHILE Elemento ; = elemento ; y
elemento ; \neq HV

En ambos

ENDDO

ENDDO

INICIAR i , j

i ← 1

j ← 1

SOLO EN 2

00. 27

Escribe " El elemento solo está
en el vector DOS "

$J \leftarrow J + 1$

SOLO EN 1

Escribe "El elemento solo está en el vecto UNO"

$I \leftarrow I + 1$

EN AMBOS

Escribe "El elemento esta en ambos
vectores "

$J \leftarrow J + 1$

$I \leftarrow I + 1$



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

LENGUAJE DE PROGRAMACION BASICO

SISTEMAS MINIMOS CUADRADOS

NOVIEMBRE, 1983

PROGRAMA PARA AJUSTE DE CURVAS POR MINIMOS CUADRADOS

CARLOS A. RAMOS LARIOS 1983

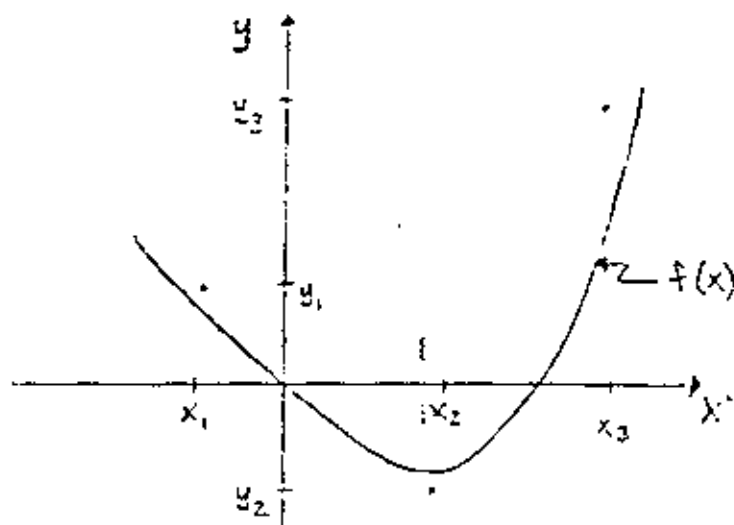
USO:

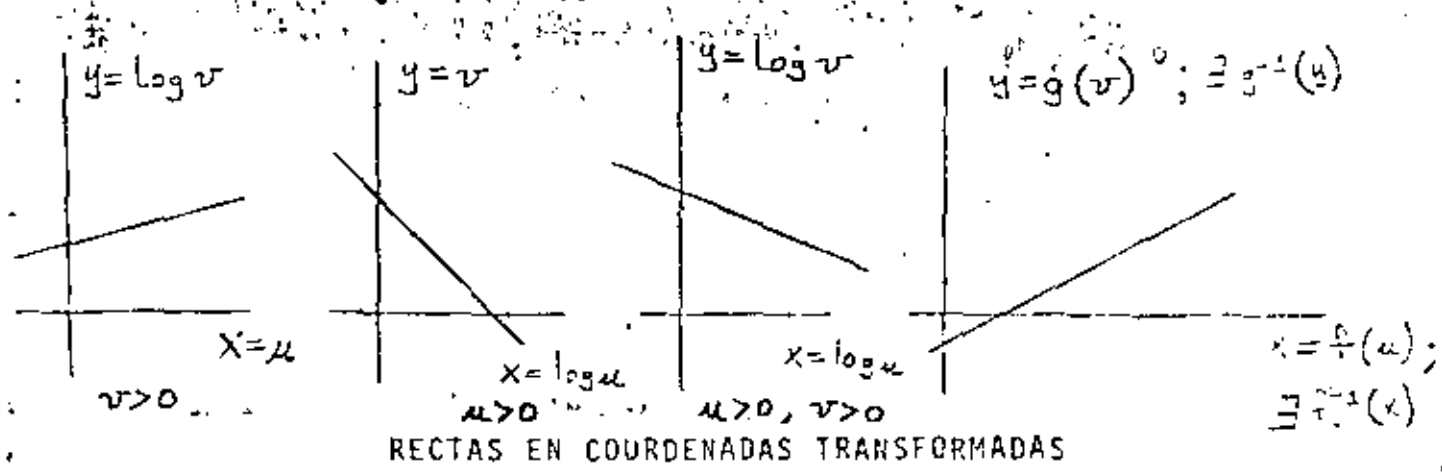
- Obtener expresiones analíticas que representen un fenómeno definido por coordenadas aisladas aplicando el criterio de los mínimos cuadrados.
- Nos limitaremos a fenómenos con una variable independiente y una variable dependiente. (X,Y)
- Las expresiones analíticas por ajustar podran ser: 1) Polinomios enteros de cualquier grado (rectas, parábolas, etc.) en las coordenadas originales, 2) Rectas en las coordenadas (X, LOGY), (LOGX, Y) (LOGX, LOGY), pudiendo utilizarse cualquier base de logaritmos, 3) Rectas con transformaciones biunivocas definidas por el usuario.

(X_i, Y_i) = Coordenadas aisladas

$f(x)$ = Curva ajustada por mínimos cuadrados

x, y = Coordenadas originales





Con las expresiones analíticas obtenidas podrán evaluarse puntos fuera de los observados (interpolación y Extrapolación)

PLANTEAMIENTO:

Sean: m puntos con abscisa diferente (datos)

n el grado máximo del polinomio que desea ajustarse (dato)

a_n, a_{n-1}, \dots, a_0 los coeficientes del polinomio (incógnitas)

$f(x) = a_n x^n + a_{n-1} x^{n-1} + \dots$ el polinomio de mejor ajuste

$E = \sum_{i=1}^m (f(x_i) - y_i)^2 w_i$ el error cuadrático a minimizar

w_i los factores de ponderación o de repetición de cada observación (dato)

Para resolver el problema, (obtener a_n, a_{n-1}, \dots, a_0), se aplican las condiciones necesarias para la existencia de un mínimo para el error

E:

$$\frac{\partial E}{\partial a_n} = 0 ; \quad \frac{\partial E}{\partial a_{n-1}} = 0 ; \quad \dots \quad \frac{\partial E}{\partial a_0} = 0 \quad (n+1 \text{ condic.})$$

Para cualquiera de estas condiciones (la jésima):

$$\begin{aligned} \frac{\partial E}{\partial a_j} &= \sum_{i=1}^m 2 \left(f(x_i) - y_i \right) w_i \left(\frac{\partial f(x_i)}{\partial a_j} \right) \\ &= 2 \sum_{i=1}^m \left(f(x_i) - y_i \right) w_i \left(x_i^j \right) \\ &= 2 \sum \left(a_n x_i^n + a_{n-1} x_i^{n-1} \dots + a_0 - y_i \right) w_i \left(x_i^j \right) \quad \left(\sum = \sum_{i=1}^m \right) \\ &= 2 \left(\sum a_n x_i^n w_i x_i^j + \sum a_{n-1} x_i^{n-1} w_i x_i^j + \dots - \sum y_i w_i x_i^j \right) \\ &= 2 \left(a_n \sum w_i x_i^{n+j} + a_{n-1} \sum w_i x_i^{n-1+j} \dots + a_0 \sum w_i x_i^j - \sum y_i w_i x_i^j \right) \end{aligned}$$

Igualando esta condición a cero, se obtiene una ecuación lineal en las incógnitas $a_n, a_{n-1} \dots a_0$:

$$a_n \sum w_i x_i^{n+j} + a_{n-1} \sum w_i x_i^{n-1+j} + \dots + a_0 \sum w_i x_i^j = \sum y_i w_i x_i^j$$

La primera condición ($\frac{\partial E}{\partial a_n} = 0$) establece que ($j = n$):

$$a_n \sum w_i x_i^{n+n} + a_{n-1} \sum w_i x_i^{n+n-1} + \dots + a_0 \sum w_i x_i^n = \sum y_i w_i x_i^n$$

La segunda condición ($\frac{\partial E}{\partial a_{n-1}} = 0$) establece que ($j = n-1$):

$$a_n \sum w_i x_i^{n+n-1} + a_{n-1} \sum w_i x_i^{n+n-2} + \dots + a_0 \sum w_i x_i^{n-1} = \sum y_i w_i x_i^{n-1}$$

Al aplicar en la misma forma todas las condiciones necesarias, se puede escribir el siguiente sistema de $n+1$ ecuaciones con $n+1$ incógnitas:

4

$$\begin{array}{ccc}
 \sum w_i x_i^{n+1} & \sum w_i x_i^{n+1-1} & \dots \\
 \sum w_i x_i^{n+1-1} & \sum w_i x_i^{n+1-2} & \\
 \vdots & & \\
 \vdots & & \\
 \vdots & & \\
 \sum w_i x_i^{n+1} & \sum w_i x_i^n & \\
 \sum w_i x_i^n & \sum w_i x_i^{n-1} & \dots
 \end{array}$$

$$\begin{array}{ccc}
 \sum w_i x_i^{n+1} & \sum w_i x_i^n & a_n \\
 \sum w_i x_i^n & \sum w_i x_i^{n-1} & a_{n-1} \\
 \vdots & \vdots & \vdots \\
 \vdots & \vdots & \vdots \\
 \sum w_i x_i^2 & \sum w_i x_i & a_2 \\
 \sum w_i x_i & \sum w_i & a_0
 \end{array}
 =
 \begin{array}{c}
 \sum y_i w_i x_i^n \\
 \sum y_i w_i x_i^{n-1} \\
 \vdots \\
 \sum y_i w_i x_i \\
 \sum y_i w_i
 \end{array}$$

De la solución del sistema anterior se obtienen los $n+1$ coeficientes a_n, a_{n-1}, \dots, a_0 .

COORDENADAS TRANSFORMADAS:

Para el caso en que se haya hecho un ajuste a una recta, ($n=1$), la expresión tiene la forma:

$$y = f(x) = a_1 x + a_0$$

Y si además los valores de (x_0, y_0) que se utilizaron para plantear el sistema de ecuaciones provienen de una transformación logarítmica sencilla o doble, es posible obtener la forma de la expresión en las coordenadas originales u y v .

CASO I:

$$y = \log_b v \quad ; \quad x = u$$

de donde

$$\log_b v = a_1 u + a_0$$

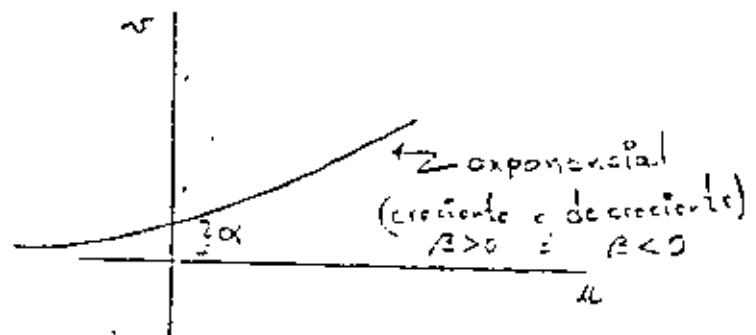
(b =base de logaritmos que se elija, $b > 0$)

$$v = b^{a_1 u + a_0}$$

$$= b^{a_1 u} \cdot b^{a_0}$$

$$= b^{a_0} \cdot b^{a_1 u}$$

$$v = \alpha \cdot b^{\beta u}$$

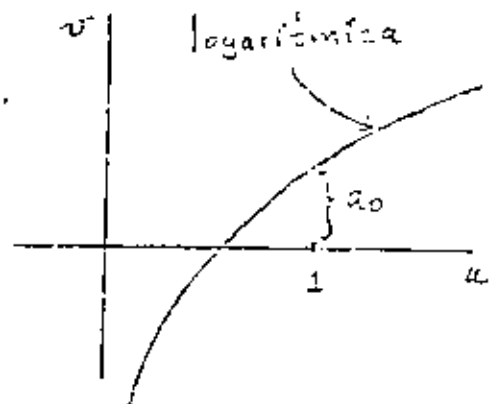


CASO II:

$$y = v ; x = \log_b u$$

de donde

$$v = a_1 \log_b u + a_0$$



CASO III:

$$y = \log_b v ; x = \log_b u$$

de donde

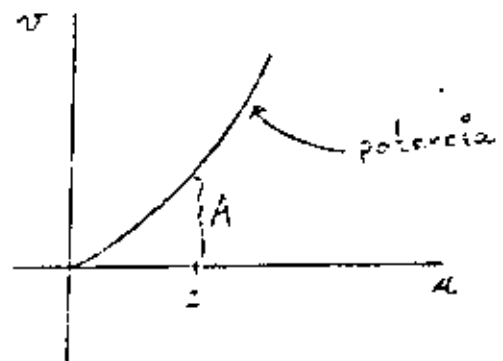
$$\log_b v = a_1 \log_b u + a_0$$

$$v = b^{a_1 \log_b u + a_0}$$

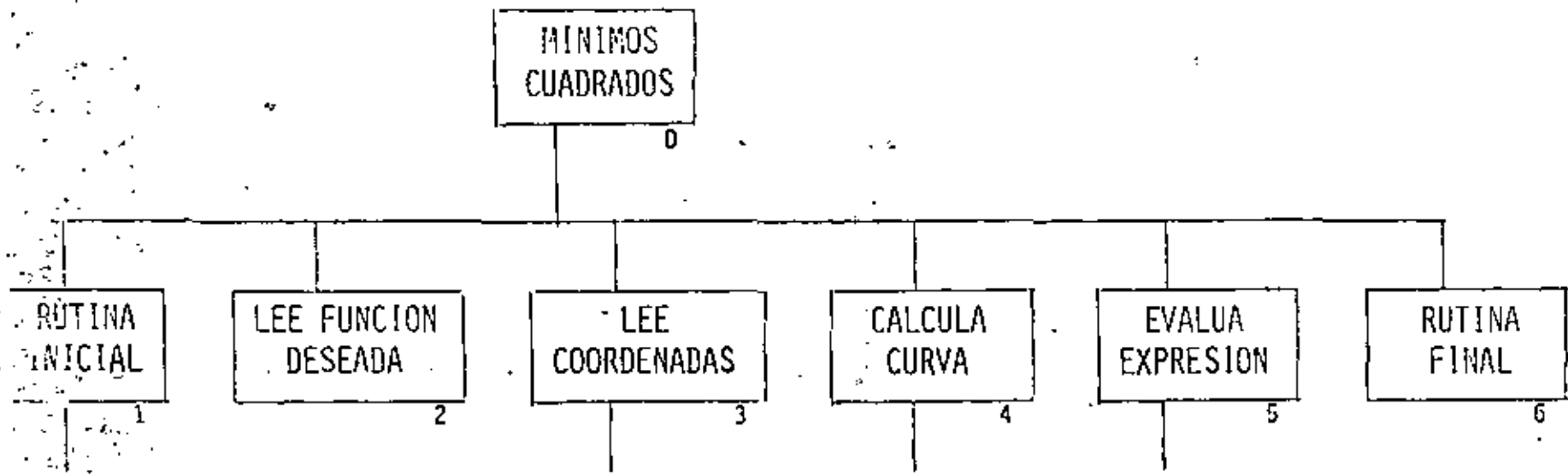
$$= b^{a_0} b^{a_1 \log_b u}$$

$$= b^{a_0} b^{\log_b u^{a_1}} = b^{a_0} u^{a_1}$$

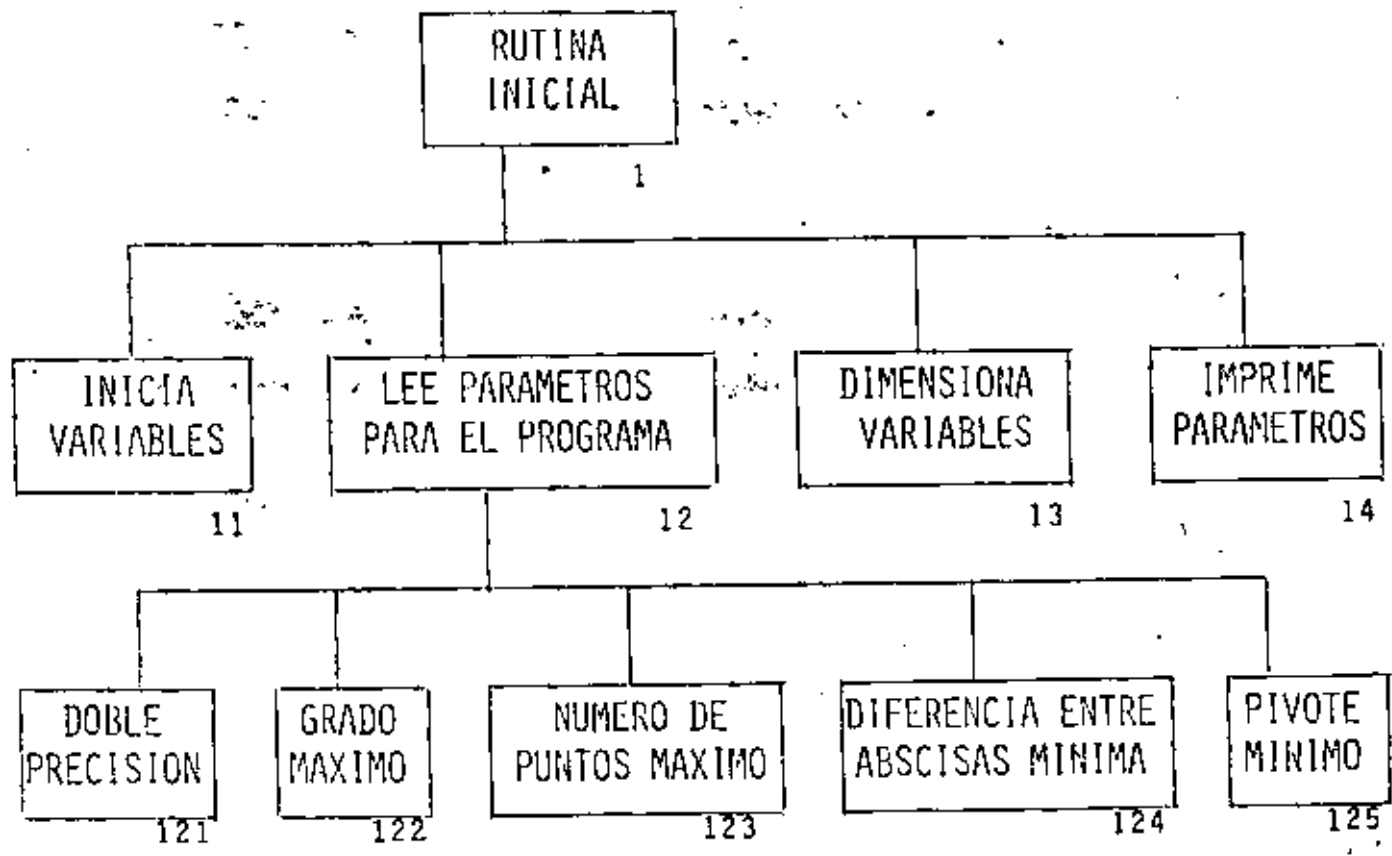
$$v = A u^B$$



Notese que B no tiene que ser necesariamente entero.



CC 7



00
00
00

LEE
COORDENADAS

3

LEE NUMERO
DE PUNTOS

31

LEE PAREJAS
Y PESOS

32

ANALIZA
PAREJAS

33

IMPRIME
PAREJAS Y PESOS

34

ORDENA POR
ABSCISA

331

CALCULA
RANGOS

332

ELIMINA
ABSCISAS
CERCANAS

333

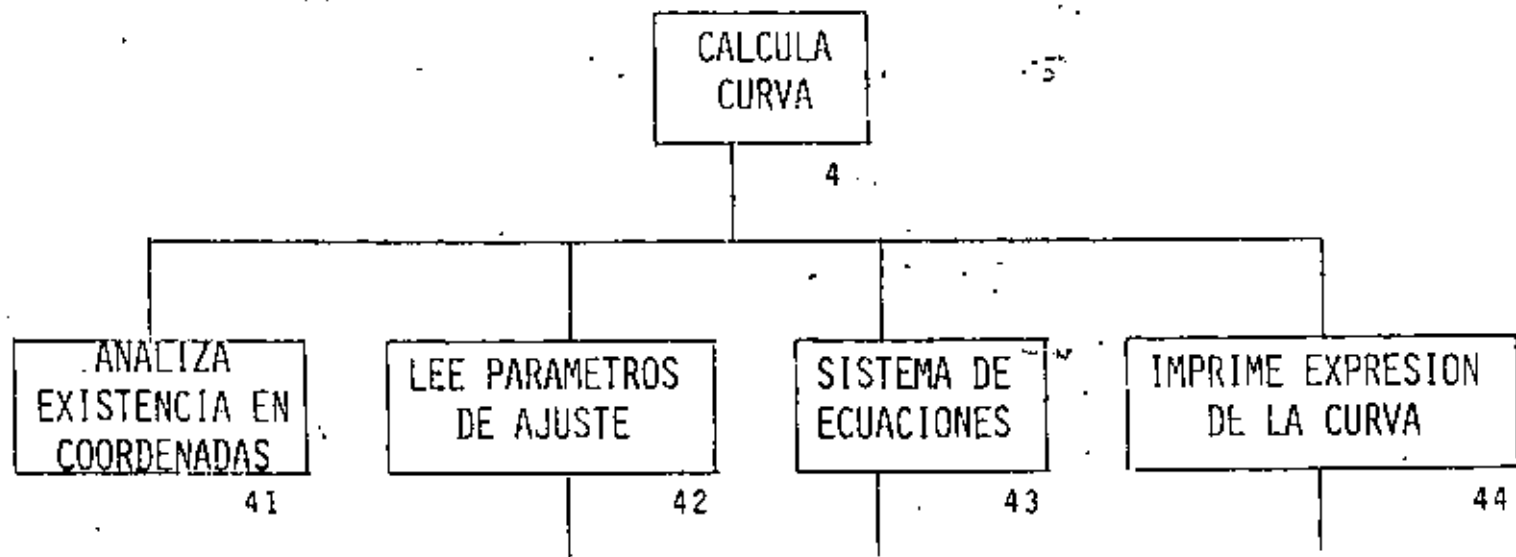
DE
X

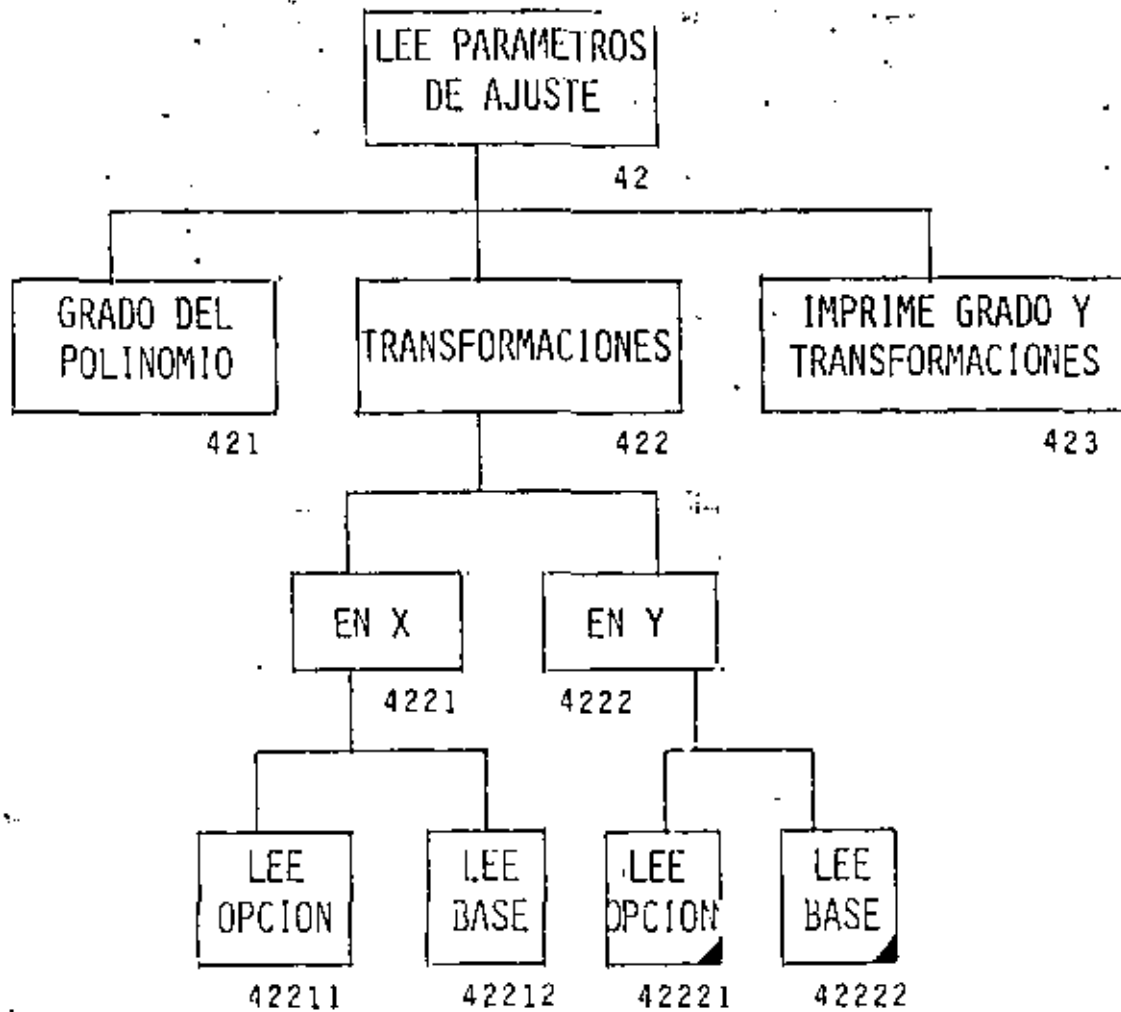
3321

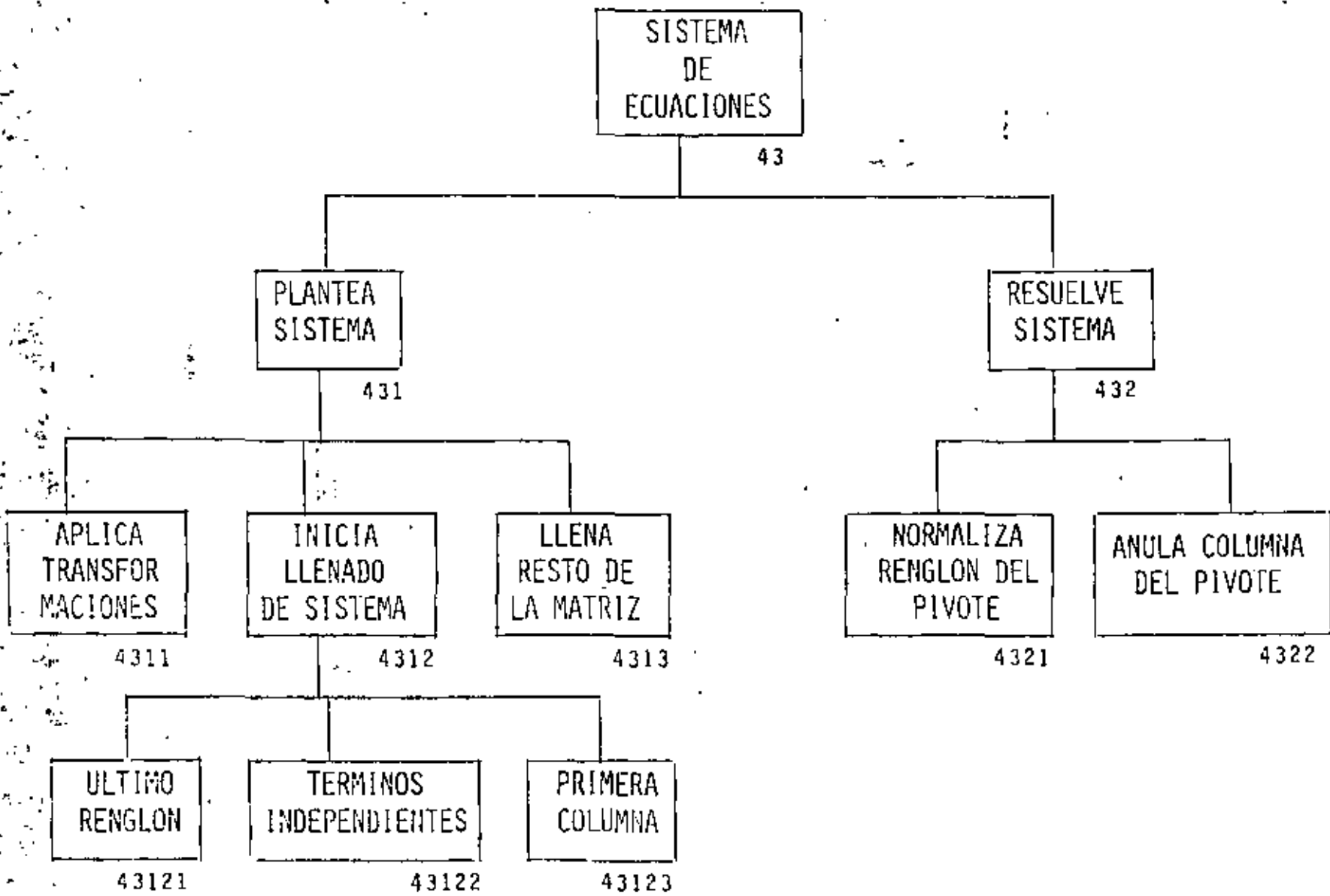
DE
Y

3322

600.9







12

IMPRIME EXPRESION
DE LA CURVA

44

COEFICIENTES
DEL POLINOMIO

441

FORMA DE LA RECTA
EN COORDENADAS
ORIGINALES

442

ERROR
CUADRATICO

443

CASO I
X-LOGY

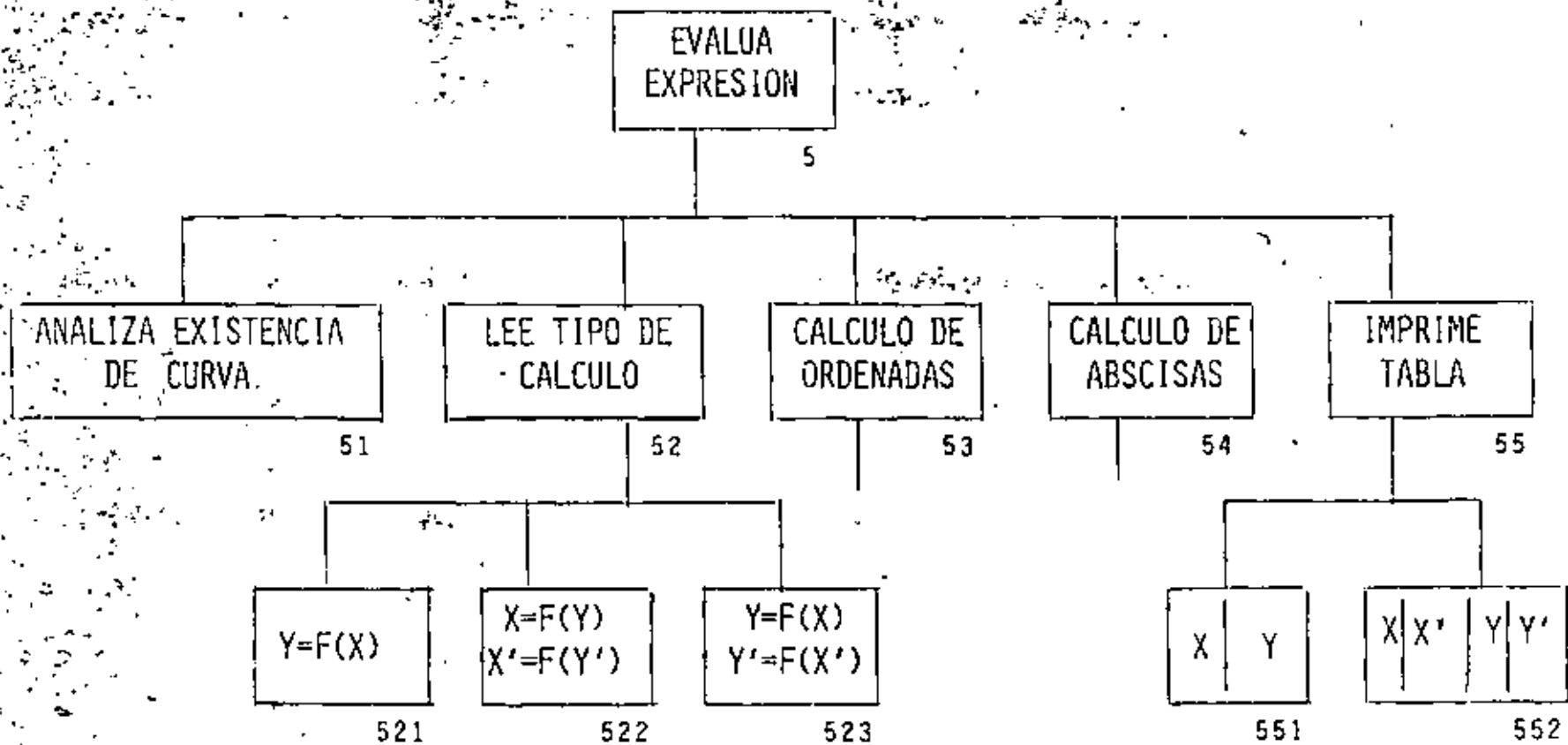
4421

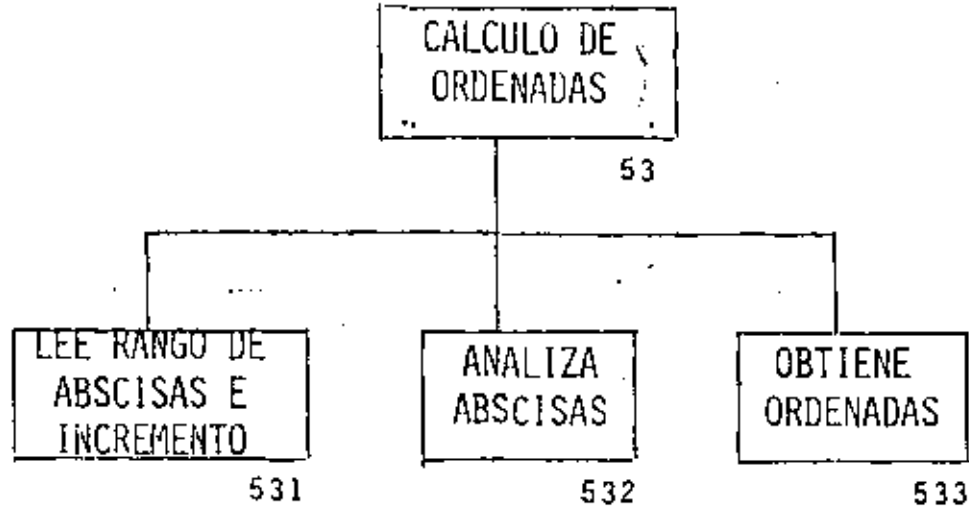
CASO II
LOG X-Y

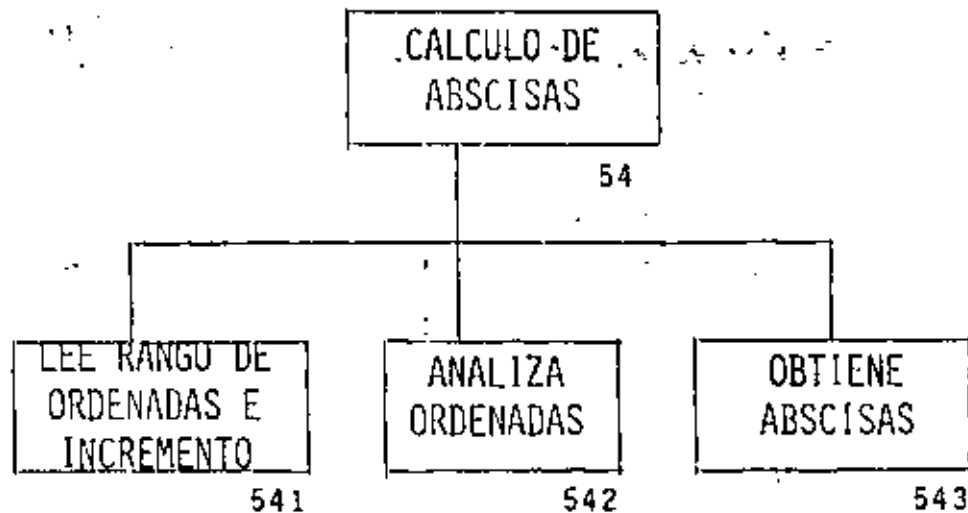
4422

CASO III
LOG X-LOGY

4423







0 MINIMOS CUADRADOS
1 RUTINA INICIAL
11 INICIA VARIABLES
12 LEE PARAMETROS PARA EL PROGRAMA
121 DOBLE PRECISION
122 GRADO MAXIMO
123 NUMERO DE PUNTOS MAXIMOS
124 DIFERENCIA ENTRE ABSCISAS MINIMA
125 PIVOTE MINIMO
13 DIMENSIONA VARIABLES
14 IMPRIME PARAMETROS
2 LEE FUNCION DESEADA
3 LEE COORDENADAS
31 LEE NUMERO DE PUNTOS
32 LEE PAREJAS Y PESOS
33 ANALIZA PAREJAS
331 ORDENA POR ABSCISA
332 CALCULA RANGOS
3321 DE X
3322 DE Y
333 ELIMINA ABSCISAS CERCANAS
34 IMPRIME PAREJAS Y PESOS
4 CALCULA CURVA
41 ANALIZA EXISTENCIA DE COORDENADAS
42 LEE PARAMETROS DE AJUSTE
421 GRADO DEL POLINOMIO
422 TRANSFORMACIONES
4221 EN X

42211 LEE OPCION
42212 LEE BASE
4222 EN Y
42221 LEE OPCION
42222 LEE BASE
423 IMPRIME GRADO Y TRANSFORMACIONES
43 SISTEMA DE ECUACIONES
431 PLANTEA SISTEMA
4311 APLICA TRANSFORMACIONES
4312 INICIA LLENADO DE SISTEMA
43121 ULTIMO RENGLON
43122 TERMINOS INDEPENDIENTES
43123 PRIMERA COLUMNA
4313 LLENA RESTO DE LA MATRIZ
432 RESUELVE SISTEMA
4321 NORMALIZA RENGLON DEL PIVOTE
4322 ANULA COLUMNA DEL PIVOTE
44 IMPRIME EXPRESION DE LA CURVA
441 COEFICIENTES DEL POLINOMIO
442 FORMA DE LA RECTA EN COORDENADAS ORIGINALES
4421 CASO I X-LOGY
4422 CASO II LOGX-Y
4423 CASO III LOGX-LOGY
443 ERROR CUADRATICO
5 EVALUA EXPRESION
51 ANALIZA EXISTENCIA DE CURVA
52 LEE TIPO DE CALCULO
521 $Y=F(X)$

522 $X=F(X) \quad X'=F(Y')$

523 $Y=F(X) \quad Y'=F(X')$

53 CALCULO DE ORDENADAS

531 LEE RANGO DE ABSCISAS E INCREMENTO

532 ANALIZA ABSCISAS

533 OBTIENE ORDENADAS

54 CALCULO DE ABSCISAS

541 LEE RANGO DE ORDENADAS E INCREMENTO

542 ANALIZA ORDENADAS

543 OBTIENE ABSCISAS

55 IMPRIME TABLA

551 X Y

552 X X' Y Y'

```

10 UNIVERSIDAD NACIONAL AUTONOMA DE MEXICO
15 FACULTAD DE INGENIERIA
20 DIVISION DE EDUCACION CONTINUA
25 SISTEMA "MINIMOS CUADRADOS"
30 CARLOS AGUSTO RAMOS LARIOS 1983
40 CLEAR
50 GOSUB 200 'ROUTINA INICIAL
60 GOSUB 1000 'LEE FUNCION DESEADA EN LA VARIABLE IN
70 IF IN=1 GOTO 160
80 ON IN-1 GOTO 70,90,110
90 GOSUB 2000 'LEE COORDENADAS
100 GOTO 130
110 GOSUB 8000 'EVALUA EXPRESION
120 GOTO 130
130 ENDCASE
140 GOSUB 1000 'LEE FUNCION DESEADA EN LA VARIABLE IN
150 GOTO 50
160 ENDDO
170 GOSUB 9000 'ROUTINA FINAL
180 END

```



```

200 'ROUTINA INICIAL
    'INICIA VARIABLES
210 CLS
220 R$='NO' : NM=5 : MM=20 : DX=0.001 : PM=0.001 :
    D2=0 : D3=0
230 'LEE PARAMETROS PARA EL PROGRAMA
240 INPUT 'SE DESEA DOBLE PRECISION SI/NO' ; R$
250 IF R$='SI' THEN
    D=1
ELSE
    D=0
ENDIF
260 INPUT 'GRADO MAXIMO DEL POLINOMIO' ; NM
270 NM=INT(NM)
280 IF NM>0 THEN GOTO 330
290 PRINT 'GRADO MAXIMO DEBE SER POSITIVO'
300 INPUT 'GRADO MAXIMO DEL POLINOMIO' ; NM
310 NM=INT(NM)
320 GOTO 280
330 ENDDO
340 INPUT 'NUMERO DE PUNTOS MAXIMO' ; MM
350 MM=INT(MM)
360 IF MM>1 THEN GOTO 410
370 PRINT 'NUMERO MAXIMO DEBE SER >1'
380 INPUT 'NUMERO DE PUNTOS MAXIMO' ; MM
390 MM=INT(MM)
400 GOTO 360
410 ENDDO
420 INPUT 'DIFERENCIA MINIMA ADMISIBLE ENTRE ABSCISAS' ; DX
430 DX=ABS(DX)
440 INPUT 'PIVOTE MINIMO ADMISIBLE AL RESOLVER SISTEMA' ; PM
450 PM=ABS(PM)
460 IF D THEN DEFDBL V-V
470 'DIMENSIONA ARREGLOS
480 DIM UX(MM),UY(MM),UR(MM),VA(NM+1),UR(NM+1,NM+1)
490 'IMPRIME PARAMETROS
500 CLS
510 PRINT 'PARAMETROS A UTILIZAR EN EL PROGRAMA:'
520 PRINT
530 IF D THEN PRINT 'SE UTILIZARA DOBLE PRECISION'
540 PRINT 'GRADO MAXIMO DEL POLINOMIO'
550 PRINT 'NUMERO DE PUNTOS MAXIMO'
560 PRINT 'DIFERENCIA MINIMA ADMISIBLE ENTRE ABSCISAS' ; DX
570 PRINT 'PIVOTE MINIMO ADMISIBLE AL RESOLVER SISTEMA' ; PM
580 INPUT 'PARA CONTINUAR PRESIONE ENTER-' ; R$
590 RETURN

```

```

1000  LEE OPCION
1010  IN=0
1020  IF IN=0 AND IN<5 GOTO 1140
1025  CLS
1030  PRINT 'SISTEMA MINIMOS CUADRADOS'
1040  PRINT
1050  PRINT 'MENU PRINCIAL:'
1060  PRINT '1.- SALIDA DEL SISTEMA'
1070  PRINT '2.- LEE COORDENADAS'
1080  PRINT '3.- CALCULA CURVA'
1090  PRINT '4.- EVALUA EXPRESION'
1100  PRINT
1110  INPUT 'INDIQUE OPCION DESEADA: ' IN
1120  IN=INT(IN)
1130  GOTO 1020
1140  'ENDDO
1150  RETURN

```

```

2000  LEE COORDENADAS
2005  CLS
2010  GOSUB 2500 'LEE NUMERO DE PUNTOS'
2020  IF M<=M1 GOTO 2060
2030  PRINT 'NUMERO DE PUNTOS ES MAYOR QUE EL MAXIMO'
2040  GOSUB 2500 'LEE NUMERO DE PUNTOS'
2050  GOTO 2020
2060  'ENDDO
2070  LEE PAREJAS Y PESOS
2080  FOR I=1 TO N
2090  PRINT 'TECLEAR VALORES DE X(';I;'), Y(';I;'), W(';I;')
2100  INPUT VX(I), VY(I), VW(I)
2110  READ UX(I), UY(I), UW(I)
2120  NEXT I
2130  'DATA
2140  'DATA
2150  'DATA
2160  'DATA
2170  'DATA
2180  GOSUB 3000 'ANALIZA PAREJAS
2190  IF M>1 THEN OZ=1
2200  IMPRIME PAREJAS Y PESOS
2210  CLS
2220  PRINT 'TABLA DE PUNTOS Y PESOS'
2230  PRINT I, 'X(I)', 'Y(I)', 'W(I)'
2240  FOR I=1 TO M
2250  PRINT I, VX(I), VY(I), VW(I)
2260  NEXT I
2265  INPUT 'PARA CONTINUAR PRESIONE ENTER: ' IN
2270  RETURN

```

```

2500 'LEE NUMERO DE PUNTOS
2510 M=0
2520 INPUT 'NUMERO DE PUNTOS EN LA TABLA' :M
2530 M=INT(M)
2540 IF M>1 THEN GOTO 2590
2550 PRINT 'NUMERO DE PUNTOS DEBE SER >1'
2560 INPUT 'NUMERO DE PUNTOS EN LA TABLA' :M
2570 M=INT(M)
2580 GOTO 2540
2590 'ENDDO
2600 RETURN
3000 'ANALIZA PAREJAS
3010 GOSUB 3500 'ORDENA POR ASCISA
3020 'ELIMINA ASCISAS CERCANAS
3030 I=1
3040 IF I=M GOTO 3190
3050 IF ABS(VX(I)-VX(I+1)) >= DX GOTO 3150
3060 PRINT 'SE ELIMINA EL PUNTO (' :VX(I+1) :', ' :VY(I+1) :')'
3070 PRINT 'POR ASCISA DEMASIADO JUNTA AL PUNTO ANTERIOR'
3080 FOR J=I+1 TO M-1
3090 VX(J)=VX(J+1)
3100 VY(J)=VY(J+1)
3110 VW(J)=VW(J+1)
3120 NEXT J
3130 M=M-1
3140 GOTO 3170
3150 'ELSE
3160 I=I+1
3170 'ENDIF
3180 GOTO 3040
3190 'ENDDO
3200 GOSUB 3800 'CALCULA RANGOS
3210 RETURN

```

```

3500 ORDENA FOR ASCISA (SORI SECUENCIAL)
3505 CLS
3510 PRINT ORDENANDO TABLA, FAVOR DE ESPERAR
3520 FOR I=1 TO M-1
3530 VZ=VX(I)
3540 FOR J=I+1 TO M
3550 IF VX(J)>=VZ THEN GOTO 3600
3560 VZ=VX(J)
3570 VX(J)=VX(I)
3580 VX(I)=VZ
3590 U1=UY(J)
3600 UY(J)=U(I)
3610 UY(I)=U1
3620 V1=VH(J)
3630 VH(J)=VH(I)
3640 VH(I)=V1
3650 NOELSE
3660 ENDIF
3670 NEXT J
3680 NEXT I
3690 PRINT FIN DEL ORDENAMIENTO
3700 RETURN
3710 CALCULA RANGOS
3720 V1=VX(1)
3730 V2=VX(M)
3740 U3=UY(1)
3750 U4=UY(1)
3760 FOR I=2 TO M
3770 IF UY(I)<U3 THEN U3=UY(I)
3780 IF UY(I)>U4 THEN U4=UY(I)
3790 NEXT I
3800 PRINT X MINIMA * U3
3810 PRINT X MAXIMA * U4
3820 PRINT Y MINIMA * U3
3830 PRINT Y MAXIMA * U4
3840 PRINT
3850 PRINT
3860 INPUT PARA CONTINUAR PRESIONE ENTER * W$
3870 RETURN

```

```

4000 CALCULA CURVA
4001 ANALIZA EXISTENCIA DE LA TABLA
4002 IF O2 THEN GOTO 4009
4003 PRINT PARA CALCULAR UNA CURVA SE REQUIEREN
4004 PRINT AL MENOS DOS PUNTOS EN LA TABLA
4005 PRINT Y SE HAN PROPORCIONADO
4006 PRINT
4007 INPUT PARA CONTINUAR PRESIONE -ENTER- :IN
4008 RETURN
4009 ENDIF
4010 GOSUB 4200 'LEE PARAMETROS DE AJUSTE
4020 GOSUB 6000 'SISTEMA DE ECUACIONES
4030 IF S THEN GOTO 4060
4040 GOSUB 7000 'IMPRIME EXPRESION DE LA CURVA
4050 GOTO 4090
4060 ELSE
4070 PRINT 'SISTEMA DE ECUACIONES SINGULAR':
4080 PRINT
4080 INPUT PARA CONTINUAR PRESIONE -ENTER- :IN
4090 ENDIF
4100 RETURN

```

```

4200 'LEE PARAMETROS DE AJUSTE
4290 'GRADO DEL POLINOMIO
4300 N=0
4310 GOSUB 4500 'LEE GRADO DEL POLINOMIO
4320 IF N<=NM GOTO 4360
4330 PRINT 'GRADO DEL POLINOMIO ES MAYOR QUE EL MAXIMO'
4340 GOSUB 4500 'LEE GRADO DEL POLINOMIO
4350 GOTO 4320
4360 ENDDO
4370 'TRANSFORMACIONES
4380 TX=0:
4390 LY=0:
4400 R=0:
4410 IF N>1 THEN GOTO 4410
4420 GOSUB 4700 'TRANSE EN X
4430 GOSUB 4900 'TRANSE EN Y
4440 ENDDO
4450 GOSUB 5500 'IMPRIME PARAMETROS DE AJUSTE
4460 RETURN

```

```

4500  'LEE GRADO DEL POLINOMIO
4510  N=0
4520  INPUT 'GRADO DEL POLINOMIO DE AJUSTE: N
4530  N=INT(N)
4540  IF N>0 THEN GOTO 4590
4550  PRINT 'GRADO DEL POLINOMIO DEBE SER >0
4560  INPUT 'GRADO DEL POLINOMIO DE AJUSTE: N
4570  N=INT(N)
4580  GOTO 4540
4590  'ENDOO
4600  RETURN

```

```

4700  'TRANSF EN X
4710  RS='X':
4720  V=V1
4720  GOSUB 5000 'PIDE TRANSF Y BASE
4730  TX=T
4740  RETURN

```

```

4900  'TRANSF EN Y
4910  RS='Y':
4920  V=V3
4920  GOSUB 5000 'PIDE TRANSF Y BASE
4930  TY=T
4940  RETURN

```

```

5000 LEER TRANSFORMACION Y BASE
5010 T=-1
5020 IF T > -1 AND T < 3 AND TR THEN GOTO 5185
5030 GOSUB 5200 LEE OPCION EN T
5040 TR=TR+T
5050 IF T < > 1 THEN GOTO 5185
5060 IF V > 0 THEN GOTO 5115
5070 PRINT 'EXISTEN VALORES NO POSITIVOS DE TR'
5080 PRINT 'POR LO QUE LA TRANSFORMACION'
5090 PRINT 'NO ES POSIBLE'
5100 INPUT 'PARA CONTINUAR PRESIONE ENTER LINEA'
5110 GOTO 5155
5115 ELSE
5120 PRINT 'BASE ACTUAL DE LOS LOGARITMOS'
5125 INPUT 'BASE DESEADA (SI HAY CAMBIO) : B'
5130 IF B > 0 THEN GOTO 5145
5135 PRINT 'LA BASE DEBE SER POSITIVA'
5140 INPUT 'BASE DESEADA : B'
5145 GOTO 5130
5145 ENDDO
5150 TR=1
5155 ENDIF
5160 GOTO 5175
5165 ELSE
5170 TR=1
5175 ENDF
5180 GOTO 5020
5185 ENDDO
5190 RETURN

```

```

5200 LEE OPCION EN T
5210 CLS:
5215 PRINT 'TRANSFORMACION EN : Rs : '
5220 PRINT '0.- NO HAY'
5230 PRINT '1.- LOGARITMICA'
5240 PRINT '2.- ESPECIAL'
5250 PRINT
5260 INPUT 'INDIQUE OPCION : T'
5270 T=INT(T)
5280 RETURN

```

```

5500  *IMPRIME PARAMETROS DE AJUSTE
5510  CLS
5520  PRINT PARAMETROS DEL AJUSTE
5525  PRINT
5529  PRINT GRADO DEL POLINOMIO
5530  IF IX THEN PRINT TRANSFORMACION EN X= ;IX
5540  IF IY THEN PRINT TRANSFORMACION EN Y= ;IY
5550  IF IX=2 OR IY=2 THEN PRINT BASE DE LOS LOGARITMOS= ;S
5560  INPUT PARA CONTINUAR OBRERA -ENTER- ;W$
5570  RETURN

```

```

6000  *SISTEMA DE ECUACIONES
6005  *PLANTEA SISTEMA
6010  GOSUB 6500  *APLICA TRANSFORMACIONES
6020  GOSUB 6800  *LLENO ULTIMO RENGLON, VECTOR Y PRIMERA COL
6030  GOSUB 6900  *LLENA RESIDUO DE MATRIZ
6050  *RESUELVE SISTEMA
6050  O3=0
6060  FOR I=1 TO N+1
6070  VP=VM(I,I)
6080  IF ABS(VP)>=PM THEN GOTO 6140
6090  PRINT EL PIVOTE DEL RENGLON ;I; ES MENOR
6100  PRINT QUE EL MINIMO ADMISIBLE
6110  INPUT PARA CONTINUAR PRESIONE -ENTER- ;W$
6120  S=0
6130  RETURN
6140  *ENDIF
6150  *NORMALIZA RENGLON DEL PIVOTE
6160  FOR J=I TO N+1
6170  VM(I,J)=VM(I,J)/VP
6180  NEXT J
6190  VA(I)=VA(I)/VP
6200  *ANULA COLUMNA DEL PIVOTE
6210  FOR J=1 TO N+1
6220  IF J=I THEN GOTO 6260
6230  VF=VM(J,I)
6240  FOR K=1 TO M+1
6250  VM(J,K)=VM(J,K)-VF*VM(I,K)
6260  NEXT K
6270  VA(J)=VA(J)-VF*VA(I)
6280  *ENDIF
6290  NEXT J
6300  NEXT I
6310  O3=1: S=1
6320  RETURN

```



```

6600 APLICA TRANSFORMACIONES
6605 EN X
6610 IF TX=0 THEN GOTO 6650
6620 IF TX=1 THEN GOTO 6690
6630 FOR I=1 TO M
6640 VR=VX(I)
6650 GOSUB 9500
6660 VX(I)=VR
6670 NEXT I
6680 GOTO 6640
6690 ELSE
6700 LB=LOG(B)
6710 FOR I=1 TO M
6720 VX(I)=LOG(VX(I))/LB
6730 NEXT I
6740 ENDIF
6750 NOELSE
6760 ENDIF
6770 EN Y
6780 IF TY=0 THEN GOTO 6760
6790 IF TY=1 THEN GOTO 6730
6800 FOR I=1 TO M
6805 VR=VY(I)
6810 GOSUB 9700
6815 VY(I)=VR
6820 NEXT I
6825 GOTO 6755
6830 ELSE
6835 LB=LOG(B)
6840 FOR I=1 TO M
6845 VY(I)=LOG(VY(I))/LB
6850 NEXT I
6855 ENDIF
6860 NOELSE
6865 ENDIF
6870 RETURN

```

```

6880 LCENA ULTIMO RENGLON, VECTOR Y PRIMERA COLUMNA
6885 FOR I=0 TO N
6890 VU=0
6894 VV=0
6898 VC=0
6900 FOR J=1 TO M
6905 VZ=VH(J)*VX(J)*XI
6910 VU=VU+VZ
6915 VV=VV+VZ*VY(J)
6920 VC=VC+VZ*VX(J)*XN
6925 NEXT J
6930 VH(N+1,N+1-I)=VU 'ULTIMO RENGLON
6935 VA(N+1-I)=VV 'VECTOR
6940 VH(N+1-I,1)=VC 'PRIMERA COLUMNA
6945 NEXT I
6950 RETURN

```

```

6900 LLENA RESTO DE MATRIZ
6910 FOR I=1 TO N
6920 FOR K=2 TO N+1
6930 VM(N+1-I,K)=VM(N+2-I,K-1)
6940 NEXT K
6950 NEXT I

```

```

7000 IMPRIME EXPRESION DE LA CURVA
7005 COEFICIENTES DEL POLINOMIO
7010 CLS
7015 PRINT COEFICIENTES DEL POLINOMIO
7017 PRINT
7020 FOR I=0 TO N
7030 PRINT VALOR DE A: I; = : VA(N+1-I)
7040 NEXT I
7050 INPUT PARA CONTINUAR PRESIONE ENTER: INS
7060 IF N=1 THEN GOSUB 7500 FORMA DE LA RECTA
7070 ERROR CUADRÁTICO
7080 E=0
7090 FOR I=1 TO H
7100 UZ=UX(I)
7105 UR=0
7110 FOR J=0 TO N
7120 UR=UR+A(N+1-J)*UZ**J
7130 NEXT J
7140 E=E+(UR-U(I)**2*UH(I)
7150 NEXT I
7160 CLS
7164 PRINT ERROR CUADRADO PONDERADO= E
7168 PRINT
7170 INPUT PARA CONTINUAR PRESIONE ENTER: INS
7180 RETURN

```

```

7500  FORMA DE LA RECTA
7510  IF TX<>1 AND TY<>1 THEN RETURN
7520  IF TX=1 THEN GOTO 7580
7525  CLS
7527  PRINT 'CASO I'
7530  PRINT 'ALEA = ' ; B * VA(2)
7540  PRINT 'BASE = ' ; B
7550  PRINT 'SEJA = ' ; VA(1)
7560  INPUT 'PARA CONTINUAR PRESIONE ENTER'
7570  GOTO 7720
7580  ELSE
7590  IF TY=1 THEN GOTO 7660
7600  CLS
7605  PRINT 'CASO II'
7610  PRINT 'A1 = ' ; VA(1)
7620  PRINT 'BASE = ' ; B
7630  PRINT 'A0 = ' ; VA(2)
7640  INPUT 'PARA CONTINUAR PRESIONE ENTER'
7650  GOTO 7710
7660  ELSE
7670  CLS
7675  PRINT 'CASO III'
7680  PRINT 'A = ' ; B * VA(2)
7690  PRINT 'B = ' ; VA(1)
7700  INPUT 'PARA CONTINUAR PRESIONE ENTER'
7710  'ENDIF'
7720  'ENDIF'
7730  RETURN

```