INTRODUCCIÓN A LA PROGRAMACIÓN EXTRUCTURADA

ETIMOLOGIA

- Desde la antigüedad la Programación se ha venido dando un interés mutuo y esto se puede ver en la actualidad teniendo como base la ampliación de las memorias centrales y las altas velocidades de los procesadores , el estilo de escritura de los programas se vuelve una de las características mas sobresalientes en la programación

4.1. Técnicas de programación.

- En esta parte las herramientas de programación son: El Diagrama de Flujo , pseudo código, etc.

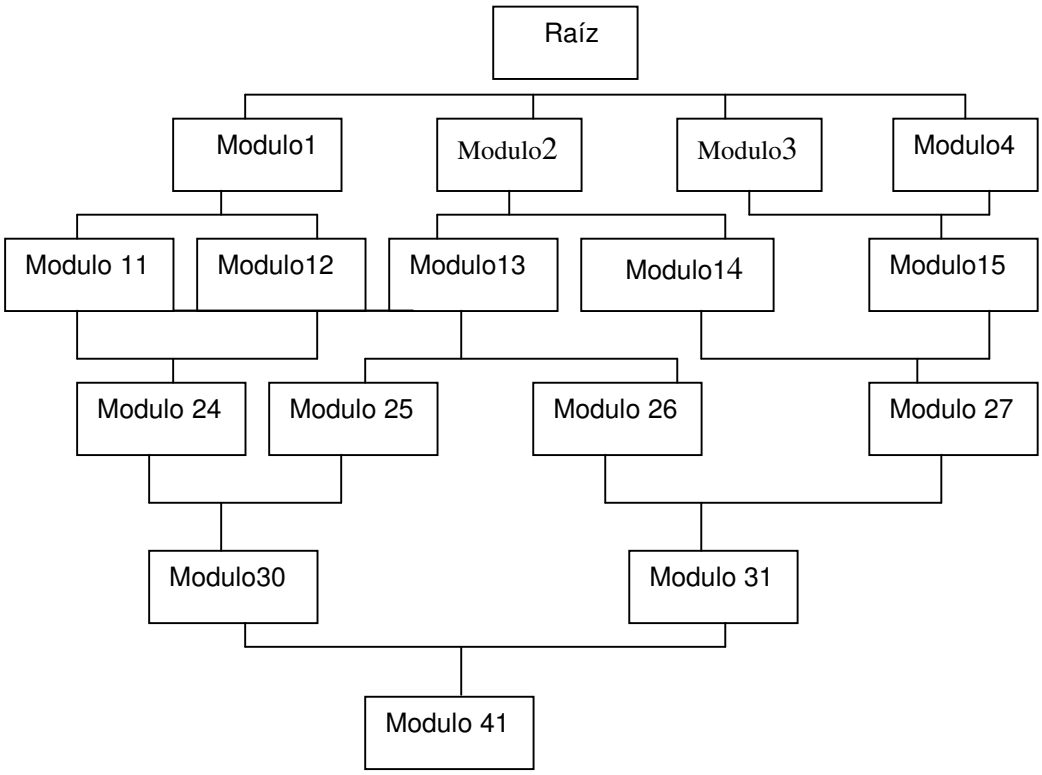
  Diagrama de Flujo: Herramienta que constituye el fundamento de Programación convencional siendo muy útil en los programas de pequeño y Mediano complejidad.

4.2. Programación modular.

Concepto: Es uno de los métodos mas flexibles y potentes para mejorar la Calidad de un programa y su productividad, en lo cual dicho programa se divide En módulos (partes independientes).

Cada programa se tiene un modulo principal denominado programa Principal que controla a los submodulos que posteriormente se denominan sub. Programas. Los módulos son independientes ya que ninguno tiene acceso Directo a otro modulo o sub. Programa, un modulo se puede modificar sin Radicalmente sin afectar a otros módulos inclusive sin alterar su funcionamiento; a este proceso se le conoce también como el método de “divide y vencerás ” (Divide and conquer).

Programación Modular



4.2.1. **Tamaño de los módulos**.

Esto se origina debido a la complejidad del programa y para un fácil entendimiento por lo que lo el tamaño mas usual de un módulo seria de una pagina (40lineas de Instrucciones).

Ejemplo: Diseñar un algoritmo que muestre los sgts resultados:

- Impresión de Cabeceras de un Informe.
- Lectura de Datos.
- Ejecutar Cálculos
- Imprimir líneas detalladas de información.
- Imprimir totales.

Solución

Modulo Principal:

- Llamada a sub. modulo"Impresión de Cabeceras".
- Llamada a sub. modulo"Proceso de Datos".
- Llamada a sub. modulo "Impresión de totales"
- Fin del Proceso

Submodulo Impresión de Cabeceras:

- Instrucción para impresión de Cabeceras.
- Retorno al modulo Principal.

Submodulo Proceso de Datos:

- Lectura de Datos.
- Ejecución de Cálculos.
- Impresión detallada de líneas.
- Retorno al modulo Principal.

Submodulo Impresión de totales:

- Instrucción de Impresión de totales.
- Retorno al modulo Principal.

4.2.2. Implementación de los módulos.

Esta implementación se realiza teniendo en cuenta el lenguaje de programación en la que se desea compilar el programa SINDO los más comunes:

a) Funciones (Lenguaje C/C++)
b) Subrutinas (Lenguaje BASICC)
c) Procedimientos (Lenguaje PASCAL)
d) Subrutinas (Lenguaje FORTRAN)
e) Secciones (Lenguaje COBOL)
f) Funciones (En todos los lenguajes C, C++, ADA, ETC)

4.3. Programación estructurada.

Concepto: Conjunto de Técnicas que han hido evolucionando, estas técnicas Empleadas aumentan considerablemente la productividad del programa reduciendo Tiempo, dinero, verificación, compilación, escritura y mantenimiento del programa

Este tipo de programación utiliza un numero limitado de estructuras de Control reduciendo a un mínimo los errores que pueden tener dicho programa .

Tipos de Programación Estructurada

- Recursos abstractos.
- Diseño descendente. (TOP-down).
- Teorema de la programación estructurada: estructuras básicas.

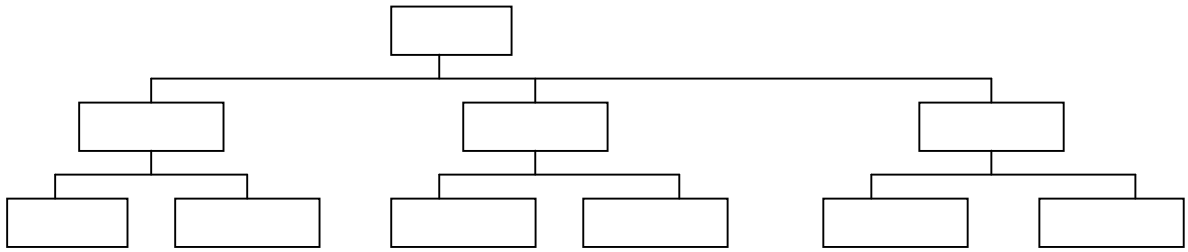
4.3.1. Recursos abstractos.

Consiste en descomponer una determinada acción compleja en función de un Numero de acciones más simples, capaces de ser ejecutadas por una Computadora y que constituirán sus instrucciones.

4.3.2. Diseño descendente. (Top-down).

Concepto: Proceso mediante el cual un problema se descompone en una serie de Niveles o pasos sucesivos de refinamiento (stepwise).
Su Metodología consiste en efectuar una relación entre las sucesivas etapas de Estructuración de modo que se relacionen unas con otras mediante entradas y Salidas de información, ósea el problema se descompone en etapas o Estructuras jerárquicas.

Diseño Descendente



4.3.3. Teorema de la programación estructurada: estructuras básicas.

Programa Propio:

- Posee un solo punto de entrada y uno de salida o fin para el control del programa.
- Existen caminos desde la entrada hasta la salida que se pueden seguir y que pasan por todas las partes del programa.
- Todas las instrucciones son ejecutables y no existen lazos o bucles infinitos (sinfín).

Un Programa propio puede ser escrito utilizando solamente tres tipos de estructuras de control:

- Estructura secuencial.
- Estructura selectiva.
- Estructura repetitiva.

4.4. Estructura secuencial.

Concepto: Es aquella en la que una acción (instrucción) sigue a otra en secuencia, Las tareas se suceden de tal modo que la salida de un programa es la entrada de Otro programa hasta llegar al final del proceso del programa.

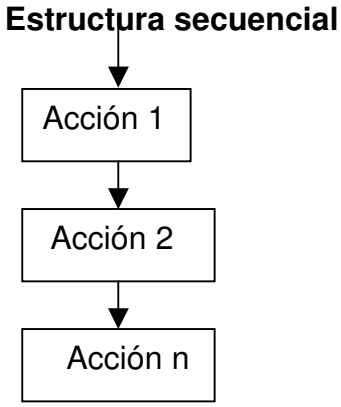
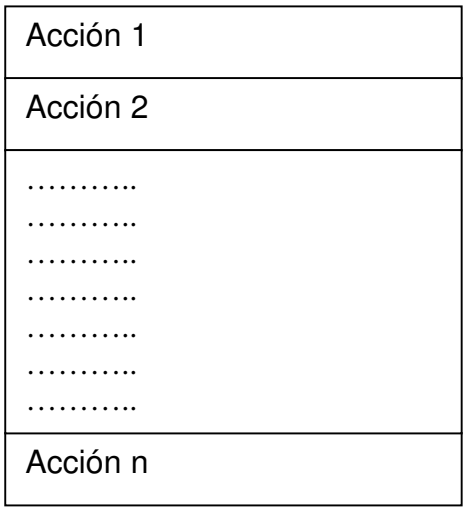


Diagrama N- S (Estructura Secuencial)



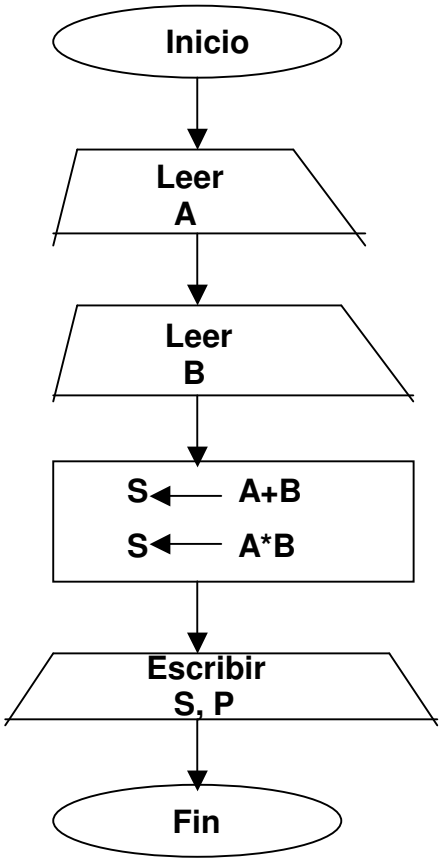
Ejemplo:
Calculo de la suma y producto de dos números. Hallar su algoritmo:

Solución

Inicio
Leer (A)
Leer (B)

S ← A+B
*P ← A*B*
Escribir(s, p)
Fin

Diagrama de Flujo



4.5. Estructura selectiva.

Función: Se utililiza para tomar decisiones lógicas; se denominan también Estructuras de decisión o alternativas, primero se evalúa primero una condición y en Función del resultado de la misma se realiza una opción u otra.
Su representación se realiza o hace con palabras de pseudo códigos (if, then, else) Español (si, entonces, si_no), con una figura geométrica en forma de rombo o bien Con un triangulo en el interior de una caja rectangular (Diagrama de Flujo)

Tipos

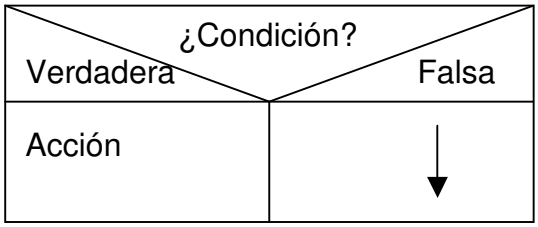
- Alternativa simple (si-entonces/if-then).
- Alternativa doble(si-entonces-si_no / if- then- else).
- Alternativa múltiple(según _sea ,caso de / case)

4.5.1 Alternativa simple (si – entonces / if – then).

Función: Ejecuta una determinada acción cuando se cumple una determinada Acción:

- Si la condición es verdadera entonces se ejecuta la acción SI (o acciones caso de ser SI una acción compuesta y constar de varias acciones)
- Si la condición es falsa , entonces no hacer nada

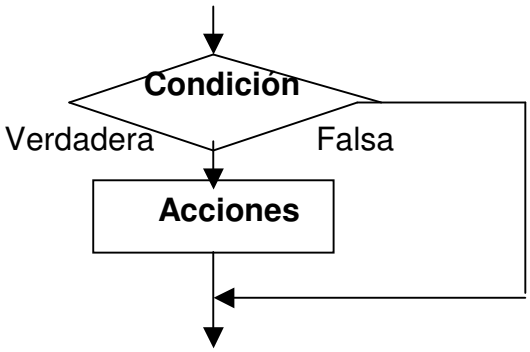
Representación Grafica de una estructura de control (Diagrama N-S)



Pseudocodigo en Castellano

S1 (acción compuesta)
Si < Condición> **entonces**
< acción s11 >
< acción s 12 >
.
.
.
Fin _ si

Diagrama de Flujo



Pseudocodigo en castellano

Si < condición > **entonces**
< acción SI >
Fin _ si

Pseudocodigo en Inglés

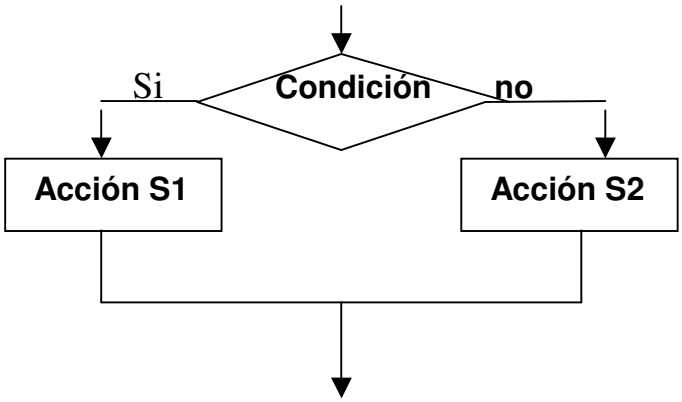
If < condición > **then**
< Acción SI >
Endif

.5.2. Alternativa doble(si-entonces-si_no / if- then- else).

Función: Ejecuta una o determinadas acciones cuando se cumple una o mas Acciones:

- Si la condición es verdadera entonces se ejecuta la acción SI (o acciones caso de ser SI una acción compuesta y constar de varias acciones).
- Si la condición es falsa, entonces se pasara a la acción sino (else); si los datos son incorrectos entonces en la acción sino se ingresara un cuadro de dialogo diciendo “datos de entrada incorrectos.

Diagrama de Flujo



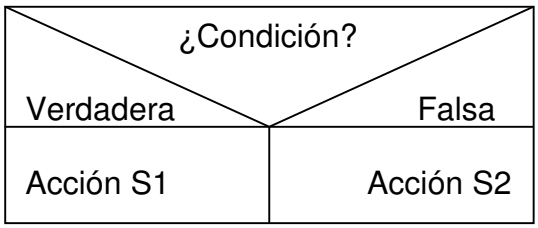
Pseudocodigo en Castellano

Si < Condición> **entonces**
< Acción SI >
Si_no
< Acción S2 >
Fin _ si

Pseudocodigo en Inglés

If < condición > **then**
< Acción SI >
Else
< Acción S2 >
Endif

Representación del Algoritmo Diagrama N-S



Pseudocodigo en Castellano

S1 (acción compuesta)
Si < Condición> **entonces**
< Acción s11 >
.
.
.
< Acción S 2n >
si_no
< Acción S21 >
< Acción S22 >
. .
< Acción S 1n >
Fin _ si

Ejemplo: Dados tres números imprimir o escribir cual es el mayor (Pseudocodigo)

Start

Read: a, b, c

If a > b and b > c **then**

m ← a

if b >a and a>c **then**

m ← b

else

m ← c

Write : m

End

4.5.3. Alternativa múltiple (según _sea, caso de / case)

Concepto: Es aquella que evalúa una expresión que podrá tomar N valores distintos (1, 2, 3, 4,…n) según que elija uno de estos valores en la condición, se realizara una de las n acciones, o lo que es igual, el flujo del algoritmo seguirá un determinado camino entre los n posibles.

Modelo (simplificado)

Según E hacer

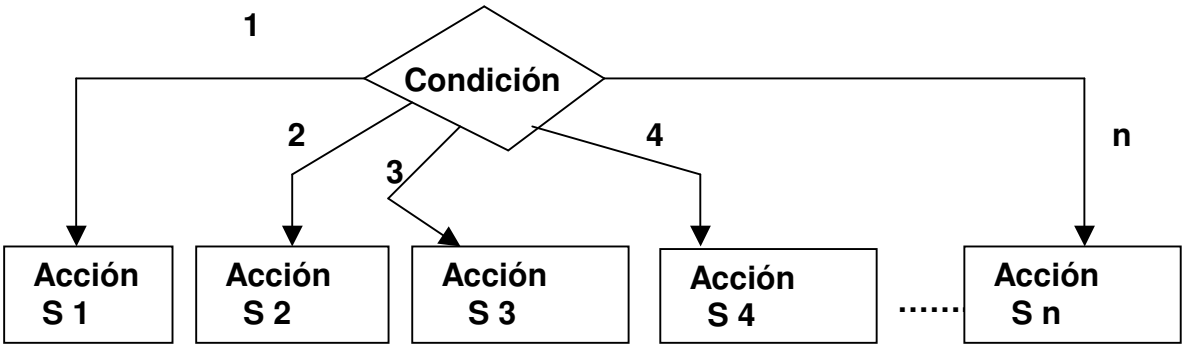
.

.

.

Fin _ según

Diagrama de Flujo



4.6. Estructura repetitiva.

Concepto: Son aquellas en las que especialmente se diseña para todas aquellas aplicaciones en las cuales una operación o conjunto de ellas deben repetirse muchas veces.

Bucles (lazos: Son estructuras que repiten una secuencia de instrucciones un numero determinado de veces.

Interacción: Es el hecho de repetir la ejecución de una secuencia de acciones; en otras palabras el algoritmo repite muchas veces las acciones.

Al utilizar un bucle para sumar una lista de números, se necesita saber cuantos números se han de sumar, para poder detenerlo en el momento preciso; las dos principales preguntas ha realizarse en el diseño de un bucle son: ¿Que contiene el bucle? y ¿Cuántas veces se debe repetir?

Casos Generales de Estructuras repetitivas

1) La condición de Salida del bucle se realiza al principio del bucle (estructura mientras)
2) La condición de Salida se origina al final del bucle; el bucle se verifica hasta que se verifique una cierta condición
3) La condición de salida se realiza con un contador que cuente el numero de interacciones. (i es un contador que cuenta desde el valor inicial (vi.) hasta el valor final (vf) con los incrementos que se consideran.)

4.6.1. Estructura mientras (“while”).

Concepto: Es aquella en que el cuerpo del bucle se repite mientras se cumple una determinada condición. Cuando se ejecuta la acción **mientras,** la primera cosa que sucede es que se evalúa la condición (una expresión booleana), si se evalúa falsa ninguna acción se tomara y el programa en la siguiente instrucción del bucle; si la expresión booleana es verdadera, entonces se ejecuta el cuerpo del bucle, depuse del cual se evalúa de nuevo la expresión booleana.

Esta expresión booleana se repite una y otra vez mientras la expresión booleana (condición) sea verdadera

Expresión de un Bucle cero veces

En una estructura mientras la primera cosa que sucede es la evaluación de la expresión booleana; si es falsa en este punto entonces el cuerpo del bucle nunca se ejecuta. Puede parecer inútil ejecutar el cuerpo del bucle cero veces, ya que no tendrá efecto en ningún valor o salida. Sin embargo no es una acción deseada

Bucles infinitos

Algunos bucles no existen fin y otros no encuentran el fin por error en su diseño, el bucle y el programa corren siempre, o al menos hasta que la computadora se apaga; en otras ocasiones el bucle no se termina nunca por que nunca se cumple la condición.

Regla Práctica

Las pruebas o tesis en las expresiones booleanas es conveniente que sean mayor o menor que en lugar de pruebas de igualdad o desigualdad. En el caso de la codificación en un lenguaje de programación, esta regla debe seguirse rígidamente el caso de comparación de números reales.

4.6.2. Estructura repetir ("repeat").

Concepto: si el valor de la expresión booleana es inicialmente falso, el cuerpo del Bucle no se ejecutara, por ello se necesitan de otros tipos de estructuras
Dicha estructura se ejecuta hasta que cumpla una condición Determinada que se comprueba hasta el final del bucle

Diferencias entre las estructuras mientras y repetir

- La estructura **mientras** termina cuando la condición es falsa, mientras que **repetir** termina cuando la condición es verdadera.
- En la estructura **repetir** el cuerpo del bucle se ejecuta siempre al menos una sola vez; por el contrario **mientras** es mas general y permite la posibilidad de que el bucle pueda no ser ejecutado. Para usar la estructura **repetir** debe estar seguro de que el cuerpo del bucle se repetirá al menos una sola vez.

4.6.3. Estructura desde/para ("for").

Concepto: Son el numero total de veces que se desea ejecutar las acciones del Bucle (numero de interacciones fijo), este ejecuta las acciones del cuerpo o del Bucle un numero especifico de veces y de modo automático controla el numero de Interacciones o pasos a través del cuerpo del bucle.

4.6.4. Salidas internas de los bucles.

Concepto: Esta nueva estructura solo esta disponible en algunos lenguajes de programación específicos la denominaremos **salir** o iterar para diferenciarla de **repetir _hasta** ya conocida .las estructuras de bucles suelen ser validas en estructuras **mientras, repetir, desde .**La estructura salir no produce un programa legible y comprensible mientras lo hacen **mientras y repetir** por lo que le recomendamos que no disponga nunca de esta estructura aunque el lenguaje a utilizar lo tenga.

4.7. Estructura de decisiones anidadas encajadas.

Concepto: Estas estructuras **si – entonces** y **si – entonces –si_no** implican la selección de una de las alternativas ya que también se puede diseñar estructuras de selección que contengan mas de dos alternativas; o sea pueden contener una u otra y así sucesivamente cualquier numero de veces , a su vez dentro de cada estructura pueden existir diferentes acciones.

4.8. Estructura repetitivas anidadas.

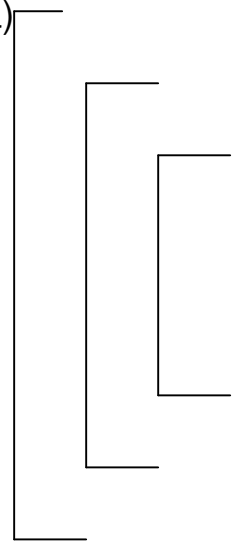
Reglas

- La estructura interna debe estar incluida debe estar incluida dentro de la externa y no puede existir solapamiento

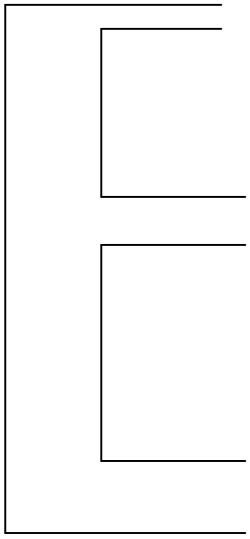
Tipos de Bucles anidados

Bucles anidados Correctos

a)

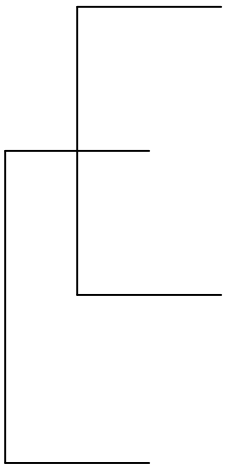


b)

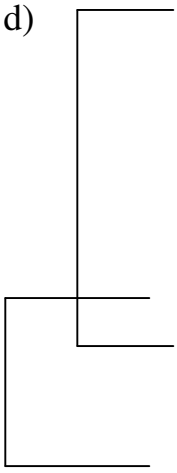


Bucles anidados incorrectos

c)



d)



Nota: Las variables índices o de control de los bucles toman valores de modo tal que por cada valor de la variable índice del ciclo externo se debe ejecutar totalmente el bucle internos .Es posible anidar cualquier tipo de estructura selectiva con tal que cumpla las condiciones de la grafica vista anteriormente

4.9. La instrucción ir_a (goto).

Aunque la instrucción ir_a (goto) la tienen todos los lenguajes de programación en su Juego de instrucciones, existen mas que dependen de ella que de otros (BASICC, FORTRAN. En general, no existe ninguna necesidad de utilizar esta instrucción, ya que esta instrucción ha sido tema de confusión y controversia por lo que los Programadores recomiendan no utilizarla en sus algoritmos y programas PASCAL- Huye de esta instrucción considerándola como nefasta y no se utiliza- ya que es más Difícil de leer y comprenderlo que otro mismo programa bien escrito, Solamente esta instrucción se recomienda utilizar en la salida de bucles.Las bifurcaciones o saltos Producidos por esta instrucción debe realizarse a programas o instrucciones que Estén enumeradas o que presenten otra referencia para el salto

EJERCICIOS

4.1. Determinar la media de una lista indefinida de números positivos, terminados con un Número negativo.

4.2. Dado el nombre o número de un mes y si el año es o no bisiesto, deducir el número de días del mes.

4.3. Sumar los números enteros de 1 a 100 mediante:

a) estructura repetir
b) estructura mientras
c) estructura desde

4.4. Determinar la media de una lista de números positivos terminada con un número no Positivo después del último número válido.

4.5. Imprimir todos los números primos entre 2 y 1000 inclusive.

4.6. Se desea leer las calificaciones de una clase de informática y contar el número total de aprobados (5 o mayor que 5)

4.7. Leer las notas de una clase de Informática y deducir todas aquellas que sean Notables (>= 7 y <9)

4.8. Leer 100 números .Determinar la media de los números positivos y la media de los Números negativos.

4.9. Un comercio dispone de dos tipos de artículos en fichas correspondientes a diversas Sucursales con los siguientes campos:

a) Código del artículo A o B
b) precio unitario del artículo
c) número de artículos
La última ficha del archivo del artículo tiene un código de artículo, una letra se pide:
a) el número de artículos existentes en cada categoría
b) el importe total de los artículos de cada categoría

4.10. Una estación climática proporciona un par de temperaturas diarias (Máxima, mínima)(no es posible que alguna o ambas temperaturas sea 9 grados).La Pareja fin de temperaturas es 0,0.Se pide determinar el número de días ,cuyas Temperaturas se han proporcionado, las medias máxima y mínima, el número de Días cuyas temperaturas se han proporcionado, las medias máxima y mínima, el Número de errores-temperaturas de oº __y el porcentaje que representaban.

4.11. Calcular:

$$E(x) = 1+x+\frac{x^2}{2!}+\ldots+\frac{x^n}{n!}$$

4.12. Calcular el enésimo término de la serie de Fibonanci definida por:
A1 = 1 A2 = 1 A3 = 1 + 2 = A1 + A2 An-1 + An-2 (n >=3)

4.13. Se pretende leer todos los empleados de una empresa__situados en un archivo empresa__ y a la terminación de la lectura del archivo se debe visualizar un Mensaje “existen trabajadores mayores de 65 años en un número de t trabajadores mayores de 65 años.

4.14. Un capital c está situado a un tipo de interés R, ¿al término de cuántos años se Doblara?

4.15. Los empleados de una fábrica trabajan en dos turnos, diurno y nocturno.Se desea Calcular el jornal diario de acuerdo con los siguientes puntos:

a) la tarifa de las horas diurnas es de 500 pesetas
b) la tarifa de las horas nocturnas es de 800 pesetas
c) caso de ser domingo, la tarifa se incrementará en 200 pesetas el turno diurno 300 Pesetas el turno nocturno.

4.16. Averiguar si dados dos números leídos del teclado, uno es divisor de otro.
4.17. Se introduce la hora del día en horas, minutos y segundos desea escribir la hora correspondiente al siguiente segundo.
4.18. Se desea conocer una serie de datos de una empresa con 50 empleados:

a) ¿Cuántos empleados ganan más de 300.000 pesetas al mes(salarios altos)
b) entre 100.000 y 300.000 pesetas (salarios medios)
c) Menos de 100.000 pesetas (salarios bajos empleados a tiempo parcial)?

4.19. Imprimir una tabla de multiplicar como

	1	2	3	4	…	15
**	**	**	**	**	…	**
1*	1	2	3	4	…	15
2*	2	4	6	8	…	30
3*	3	6	9	12	…	45
4*	4	8	12	16	…	60
.						
.						
.						
15*	15	30	45	60	…	225

4.20. Dado un entero positivo n(>1),comprobar si es primo o completo .

CONTENIDO

4.1. Técnicas de programación.

4.2. Programación modular.

4.3. Programación estructurada.

4.4. Estructurura secuencial.

4.5. Estructura selectiva.

4.6. Estructura repetitiva.

4.7. Estructura de decisión anidada.

4.8. Estructura repetitivas anidadas.

4.9. La instrucción ir_a (goto).

Bibliografía

Luis joyanes Aguilar MC Graw Hill “Fundamentos de Programación” Algoritmos Y Estructura de Datos -Segunda Edición – Págs. 714 (95- 161_ capitulo 4-)