

Diagrama de flujo Nassi-Shneiderman

Introducción a la programación

Da Vinci
PRIMERA ESCUELA DE ARTE MULTIMEDIAL



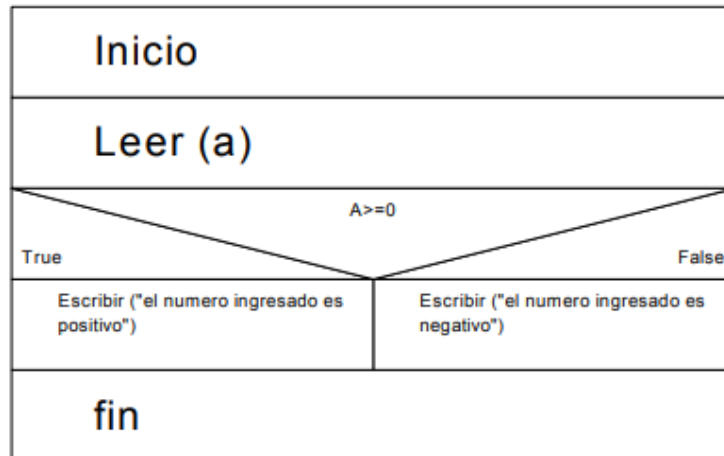
Tabla de contenido

Concepto.....	2
Representación.....	2
Estructura de asignación	3
Estructura secuencial.....	3
Estructuras selectivas o de decisión	4
Estructuras repetitivas.....	6
Estructura Repetitiva FOR (Para).....	6
Estructura Repetitiva WHILE (Mientras).....	7
Estructura Repetitiva Repeat (Repetir)	8
Funciones.....	9
Procedimientos.....	10

Concepto

El diagrama N-S de Nassi-Shneiderman (conocido también como Chapin) es como un diagrama de flujo en el que se omiten las flechas de unión y las cajas son contiguas. Las acciones sucesivas se escriben en cajas sucesivas y como en los diagramas de flujo se pueden escribir diferentes acciones en una caja.

Ejemplo de diagrama de Nassi-Shneiderman (N-S):



Representación

Para hacer un diagrama de flujo necesitarás definir el propósito y el alcance, ordenar las tareas cronológicamente, después organizarlas por tipo y símbolo de diagrama de flujo, dibujar el diagrama y perfeccionarlo. Independientemente de que escribas el diagrama de flujo en un cuaderno o de que uses una plantilla para trazar un proceso oficial, tendrás que seguir estos cinco pasos para generar un diagrama de flujo útil y razonable:

La representación en este diagrama se realiza a través de los siguientes componentes:

- I. Declaración de variables, constantes e inicialización.
- II. Ingreso de valores
- III. Proceso de valores con el objetivo de obtener resultados en el que se puede combinar diferentes componentes (Estructuras de Asignación, Estructuras Secuenciales, Selectivas o de Decisión, Repetitivas)
- IV. Salida de resultados obtenidos

El primer componente consiste en una caja en la cual se van a declarar todas las variables y/o constantes que se utilizarán en el resto del algoritmo a las cuales se le asigna un valor inicial.

$v1 \leftarrow 0, v2 \leftarrow 0, c1 \leftarrow 0$

El ingreso de valores se representa escribiendo dentro del bloque la palabra leer y entre paréntesis el valor que se ingresa.

Leer(A)

Estructura de asignación

Las operaciones de asignación son el modo de darles valores a una variable. La operación de asignación se representa con el símbolo u operador \leftarrow .

El formato general de una operación de asignación es:

Nombre de la variable \leftarrow expresión

Ejemplos:

- $A \leftarrow 5$
- $total \leftarrow 3+6$
- $N \leftarrow 0$
- $N \leftarrow N+1$
- $M \leftarrow 8<5$
- $Y \leftarrow 'CALLE'$

Los tipos de asignación pueden ser:

- **Asignación simple.** - cuando a una variable le corresponde nada más que un dato ($A \leftarrow 5$)
- **Asignación aritmética-** cuando a una variable le corresponde el resultado de una expresión aritmética ($total \leftarrow 3+6$)
- **Asignación lógica.** - cuando a una variable le corresponde el resultado de una expresión lógica ($M \leftarrow 8<5$)

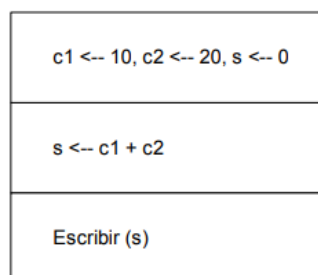
Estructura secuencial

Se caracterizan porque una acción se ejecuta detrás de otra. El flujo del programa coincide con el orden físico en el que se ha ido poniendo las instrucciones.

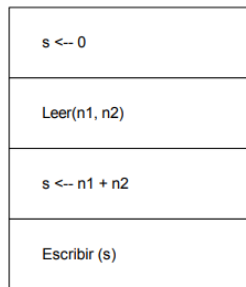


Ejemplos:

- I. Declarar dos constantes e inicializarlas con los valores 10 y 20 respectivamente, sumarlas y mostrar el resultado.




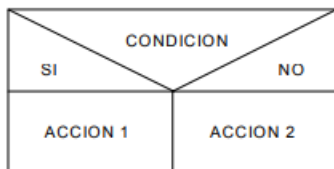
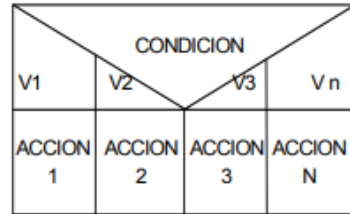
- II. Declarar dos constantes e inicializarlas pero que los valores los ingrese el usuario, luego sumarlos y mostrar el resultado.



Estructuras selectivas o de decisión

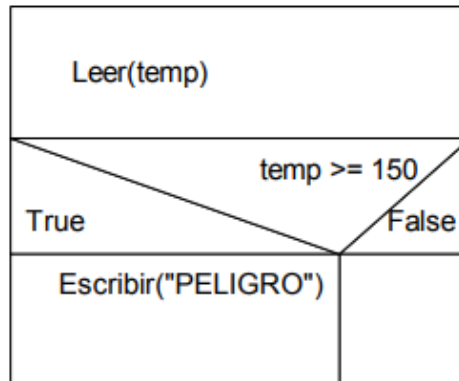
Para la aplicación se debe tener en cuenta las expresiones lógicas cuyo valor es verdadero o falso, se denomina también expresiones booleanas.

En conclusión, la estructuras selectivas o decisión o comparación o pregunta, se ejecutan unas acciones u otras según se cumpla o no una determinada condición; pueden ser: simples, dobles, o múltiples.

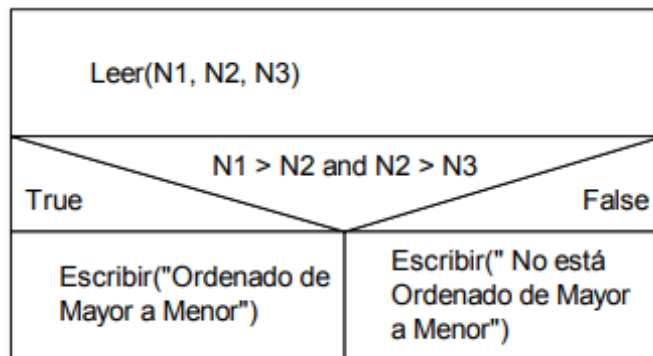
Simple	Dobles	Múltiples
<p>Simple. - Se evalúa la condición y si esta da como resultado verdad se ejecuta una determinada acción o grupo de acciones, en caso contrario se salta dicho grupo de acciones.</p>  <p>SI <condición> entonces <Acción o acciones> fin_si</p>	<p>Dobles. - Cuando el resultado de evaluar la condición es verdad se ejecuta una determinada acción o grupo de acciones y si el resultado es falso otra acción o grupo de acciones diferentes.</p>  <p>SI <condición> entonces <Acción 1> SI_NO <Acción 2> fin_si</p>	<p>Múltiples. - Se ejecutan unas acciones u otras según el resultado que se obtenga al evaluar una expresión.</p> 

Ejemplos

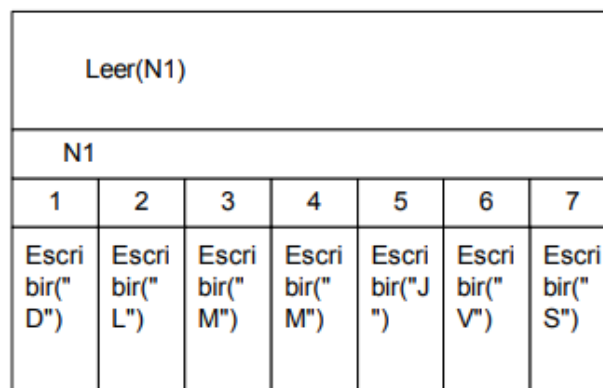
- 1- La temperatura crítica de una caldera es 150 grados centígrados. Determinar e informar mediante el mensaje PELIGRO si la temperatura medida en un momento dado es mayor o igual que la crítica.



- 2- Determinar si un conjunto de tres valores numéricos dados que están ordenados de mayor a menor.



- 3- Ingresar un número del 1 al 7 y devolver el día de la semana correspondiente al mismo.



Estructuras repetitivas

Las estructuras que repiten una secuencia de instrucciones un número determinado de veces o denominadas bucles, y se llama iteración al hecho de repetir la ejecución de una secuencia de acciones.

Las dos principales preguntas a realizarse en el diseño de un bucle son:

1. ¿Qué contiene el bucle?
2. ¿Cuántas veces se debe repetir?

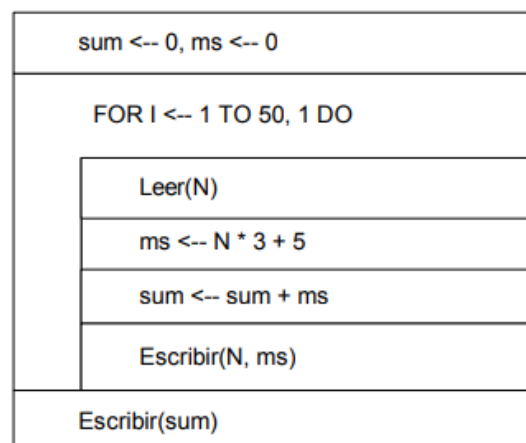
Estructura Repetitiva FOR (Para)

En muchas ocasiones se conoce de antemano el número de veces que se desea ejecutar las acciones de un bucle. En estos casos en los que el número de iteraciones es fijo se deberá usar la estructura FOR (para)

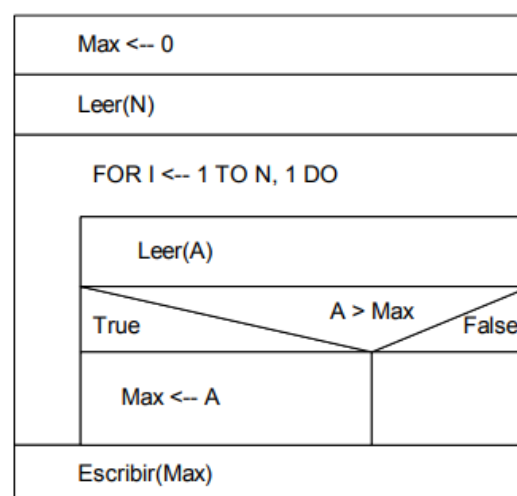
La estructura FOR ejecuta las acciones del cuerpo de un bucle un número especificado de veces y de modo automático controla el número de iteraciones o pasos a través del cuerpo del bucle.

Ejemplos.

1- Leer sucesivamente 50 valores numéricos. A cada valor multiplicarlo por tres y sumarle 5. Informar el resultado de dicha expresión junto al número que lo origina. Al final exhibir el valor acumulado de los 50 valores calculados.

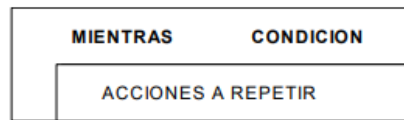


2- Hallar e informar el máximo de un conjunto de N números (N \geq 2)



Estructura Repetitiva WHILE (Mientras)

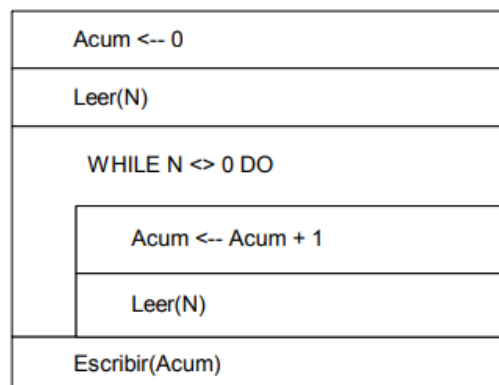
La estructura repetitiva WHILE (mientras o hacer mientras) es aquella en que el cuerpo del bucle se repite mientras se cumple una determinada condición. Cuando se ejecuta la instrucción WHILE, lo primero que se verifica es la condición (una expresión booleana). Si se evalúa falsa, el programa finaliza el bucle y sigue la instrucción a continuación en el cuerpo del algoritmo, si la expresión booleana es verdadera, entonces se ejecuta el cuerpo del bucle, y de nuevo se evalúa la expresión booleana, este proceso se repite una y otra vez mientras el resultado de la expresión booleana (condición) sea verdadera.



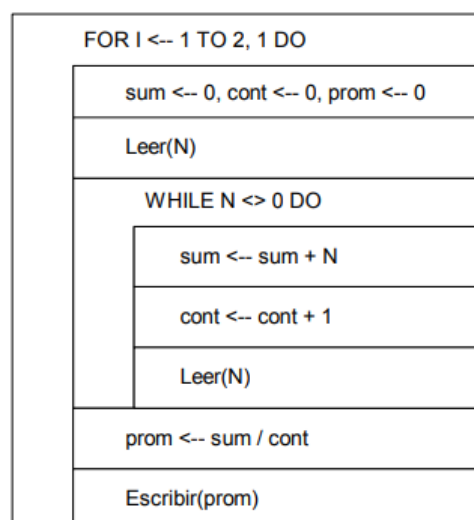
MIENTRAS condición **HACER**
 <Acción 1>
 <Acción 2>

 <Acción N>
FIN_MIENTRAS

Ejemplos 1- Se dispone de un conjunto de números distintos de cero salvo el último valor. Determinar e informar la cantidad de números que lo forman.



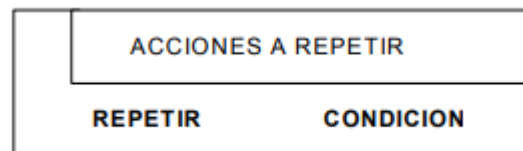
2- Se tiene un conjunto de números formados de la siguiente manera: primero todos los positivos, luego un valor nulo, a continuación, todos los negativos y finalmente otro valor nulo. Calcular y exhibir el promedio de los valores positivos y el promedio de los negativos.



Estructura Repetitiva Repeat (Repetir)

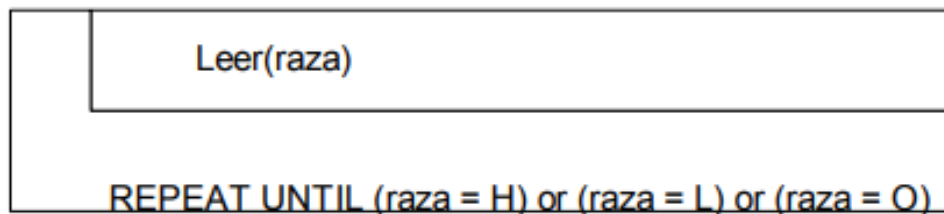
Existen muchas situaciones en las que se desea que un bucle se ejecute por lo menos una vez antes de comprobar la condición de repetición. En la estructura mientras si el valor de la expresión booleana es inicialmente falso, el cuerpo del bucle no se ejecutará; por ello se necesitan otros tipos de estructuras repetitivas. La estructura Repeat (repetir) se ejecuta hasta que se cumpla la una condición determinada que se comprueba al final del bucle.

Este bucle se repite siempre y cuando el valor de la condición booleana sea falso, lo opuesto de la sentencia WHILE.

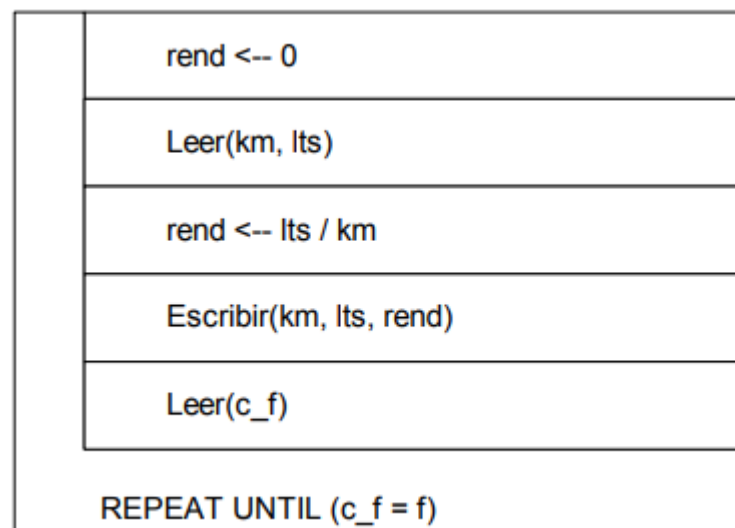


Ejemplos

- 1- Validar que una raza ingresada sea H, L u O.



- 2- Se leen pares de valores. El primero es la cantidad de kilómetros recorridos y el segundo el consumo de nafta. Para cada par calcular el rendimiento (litros / kilómetros). Emitir un listado con título y valor. Indicar si se introducirá un nuevo par de datos o se dará por finalizado el proceso.



La salida se representa escribiendo dentro del bloque la palabra **escribir** y entre paréntesis lo que se quiere mostrar (o sea el resultad obtenido)

Escribir(A)

Funciones

Una función es un grupo de tareas dentro de un programa que forman un número determinado de operaciones sobre un conjunto de argumentos dados y devuelve UN SOLO VALOR. Cada vez que es llamada una función, se transfiere el control al bloque de operaciones definidas por esta función. Después de que las operaciones han sido ejecutadas, el control vuelve al programa donde fue llamada la función. La invocación de una función es de la forma:

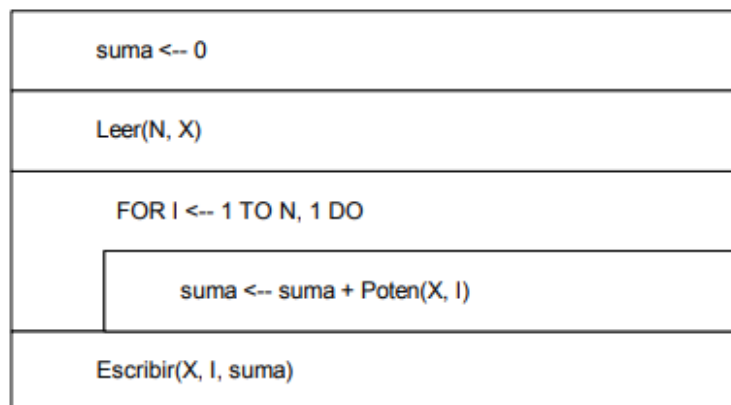
nombre (argumento1, argumento2, ...)

donde **nombre** es el nombre de la función y donde devuelve el valor y cada argumento puede ser cualquier variable válida, constante o expresión.

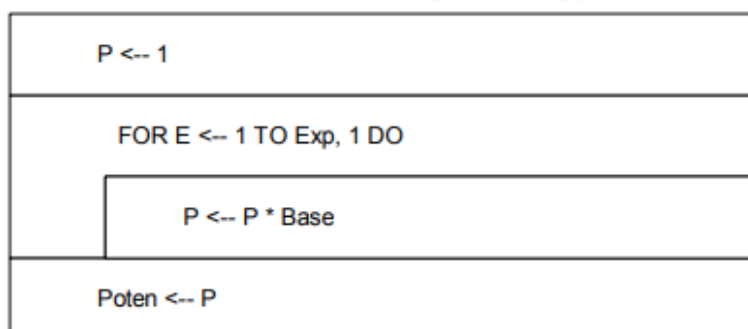
El orden de la lista de argumentos en la llamada a la función es el mismo que el orden que tienen los argumentos en la definición de la función. Si una función no necesita de argumentos, estos se omiten de la declaración.

Ejemplo.

Escribir un programa que calcule la expresión $\sum x^i$, sumatoria de N elementos. Para evaluar cada uno de los términos de la sumatoria, crear y utilizar una función llamada Poten, que tenga como argumentos la base x y el exponencial i. Exhibir: x, n y el resultado de la sumatoria.



FUNCTION Poten(Base, Exp)



Procedimientos

Los procedimientos son subprogramas similares a las funciones, pero con dos diferencias importantes.

La llamada a un procedimiento es igual a la de la función:

nombre (argumento1, argumento2, ...)

La primera diferencia es que la llamada debe estar sola, es decir, no puede estar formando parte de expresiones, ni asignada a una variable, ni en un while, ni en un if.

La segunda es que con el procedimiento no se devuelve un solo valor al punto de llamada, sino que se puede devolver cualquier número de valores por medio de los argumentos. La devolución de valores se produce a través de los argumentos.

Ejemplo

1- Realizar el ejemplo anterior, pero en lugar de la función llamar a un procedimiento de nombre Poten.

