UNIDAD 2.- ESTRUCTURA ELEMENTAL DE DATOS. DISEÑO DE ALGORITMOS.

OBJETIVOS:

- Comprender las nociones básicas para la construcción de algoritmos, la forma de realizar cálculos y la noción de acción.
- Formular y resolver problemas, diseñando las estrategias correspondientes de manera clara, sistémica y por sobre todo sencilla, mediante el diseño de algoritmos.

TEMAS:

1. Contadores, acumuladores y banderas.
2. Estructuras de Repetición.

1. CONTADORES, ACUMULADORES, BANDERAS

Todas éstas son tipos de variables.

CONTADOR

Cuenta vueltas o ciclos de repetición para poder comparar con los ciclos que se realizan. Posee en general la siguiente sintaxis:

IDENTIFICADOR = IDENTIFICADOR + CONSTANTE

C = C + 1

ACUMULADOR

Suma una determinada cantidad de valores. Sirve para acumular valores en forma progresiva, reteniendo siempre el último valor o resultado de ese proceso de acumulación. Posee en general la siguiente sintaxis:

IDENTIFICADOR = IDENTIFICADOR + VARIABLE

S = S + X

BANDERA

Es aquella que el programador utiliza para hacer posible el proceso de la solución en aquellos casos que sea necesario desviar convenientemente el flujo de la solución.

En general se le da un valor inicial, el cual cambia después de una determinada condición:

IDENTIFICADOR = CONSTANTE (VALOR INICIAL)

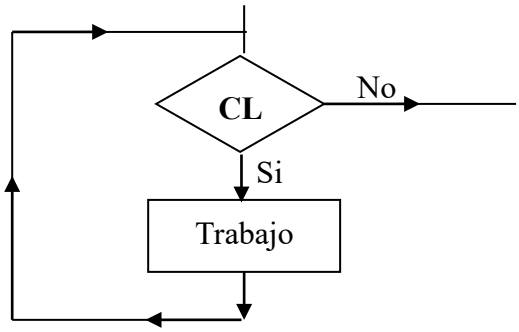
2. ESTRUCTURA DE REPETICIÓN O ITERACIÓN

El manejo de la vida diaria nos hace repetir determinadas operaciones, como por ejemplo para encontrar el resultado de la multiplicación de 2x3, se debe sumar 2 veces el número 3 ó 3 veces el número 2. Es en estos casos en los cuales tienen su aplicación estas estructuras, que son las que trataremos en el presente trabajo práctico.

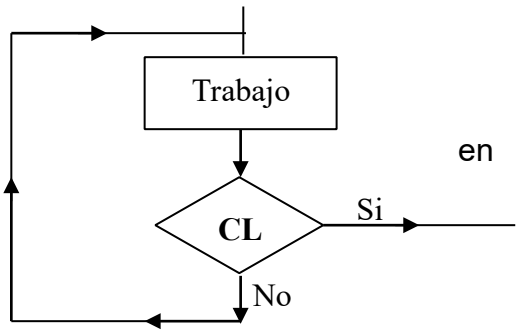
Las estructuras de iteración y las reglas de funcionamiento de cada una de ellas son:

1. *Estructura While do*
2. *Estructura Repeat Until*

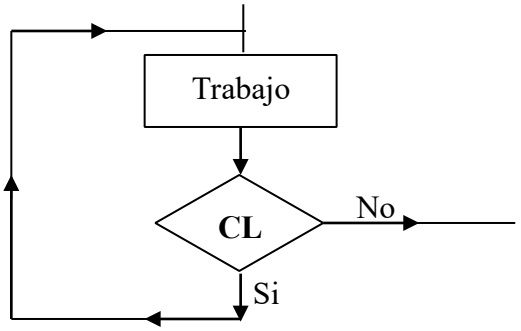
1. ***Estructura While***
   - La expresión lógica se evalúa antes de la ejecución del bucle. Si la condición es verdadera se ejecuta las acciones incluidas en el bucle, caso contrario el control pasa a la sentencia siguiente al lazo.
   - Si la expresión lógica no se cumple cuando se ejecuta el bucle por primera vez, el conjunto de acciones del lazo no se ejecuta nunca.
   - Mientras la expresión lógica se cumpla el bucle se ejecuta.



2. ***Estructura Repeat Until***
   - La expresión lógica se evalúa luego de ejecutarse todas las sentencias del bucle.
   - Si la expresión lógica es falsa, se repite el bucle ejecutándose nuevamente todas las acciones incluidas él.
   - Si la expresión lógica es verdadera, se interrumpe la ejecución del bucle y se transfiere el control a la sentencia siguiente a Until.



Esta estructura tiene su correspondiente en Lenguaje C, la estructura ***Do while***. Con la diferencia que en ella siempre se ejecuta el trabajo y se repite el mismo si la condición es verdadera**.**



Dentro de los problemas que se pueden resolver con estas estructuras encontramos:

1. Problemas donde se conoce la cantidad de veces que se repite el ciclo. Generalmente N.

2. Problemas donde NO se conoce la cantidad de veces que se repite el ciclo.
3. Problemas de generación de valores.

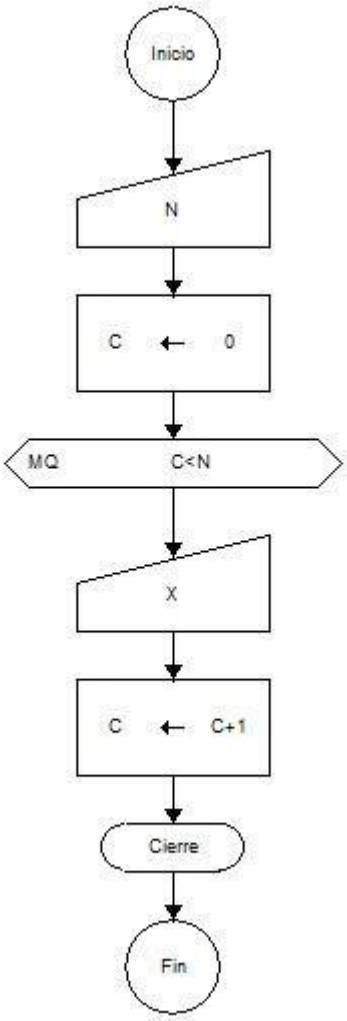
1. PROBLEMAS DONDE SE CONOCE LA CANTIDAD DE VECES QUE SE REPITE EL CICLO.

Ejemplo 1:
Introduzca N números enteros.

Se tratará de determinar en este problema cuáles son los datos y cuáles los resultados. En este caso particular se poseen datos, pero no resultados, ya que no se especifica un trabajo con ellos.

> **Para tener en cuenta**: Cuando se utiliza en el enunciado de un problema alguna de las siguientes palabras: “introduzca”, “ingrese”, o “cargue”, se hace referencia a los datos de este; mientras que cuando se expresa “muestre”, “averigüe”, “dé a conocer”, “obtenga”, o “se necesita saber”, estamos señalando cuáles son los resultados requeridos.

Datos
N: identificador que indica la cantidad de números a Ingrese. X: identificador que indica a cada uno de los números.



Ejemplo 2:
Introduzca N números enteros y muestre la suma de estos.

Datos
N: identificador que indica la cantidad de números a ingresar. X: identificador que indica a cada uno de los números.

Resultado
S: identificador que indica la suma de los valores introducidos.

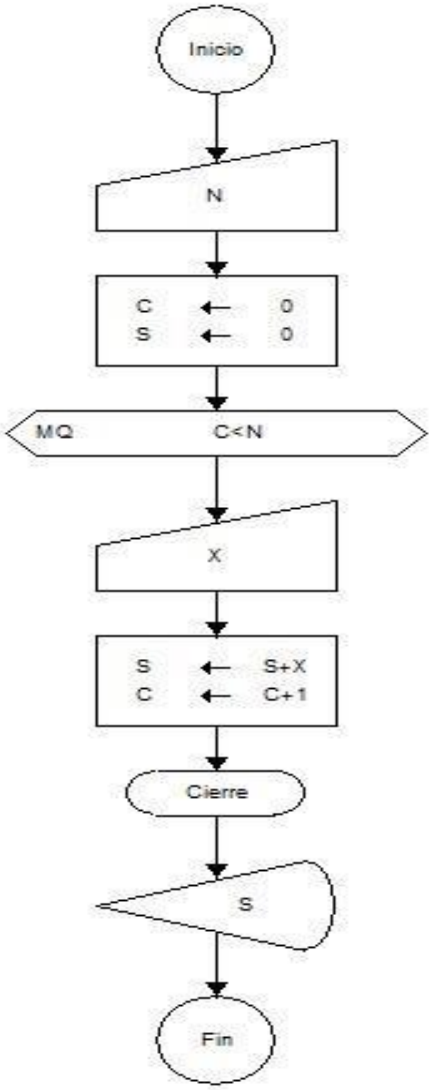
Cabe señalar que al diagrama del punto anterior lo único que se agrega es una variable acumuladora.

Variable ACUMULADOR:

Este tipo de variable debe ser inicializada con el valor neutro de la operación en la que intervenga:
0 (cero) si se intenta sumar
1 (uno) si se pretende acumular sucesivos productos.

Expresiones generales en la que interviene:

- *S=S+X*, donde *S* es la variable acumuladora de la suma y X es la variable numérica que modifica el valor de *S*.
- *F= F*X* donde F es la variable acumuladora del producto y X es la variable que modifica el valor de F.



Ejemplo 3:
Introduzca N números enteros y muestre el promedio de estos.

Datos
N: identificador que indica la cantidad de números a ingresar. X: identificador que indica a cada uno de los números.

Resultado
P: identificador que indica el promedio de los valores introducidos.

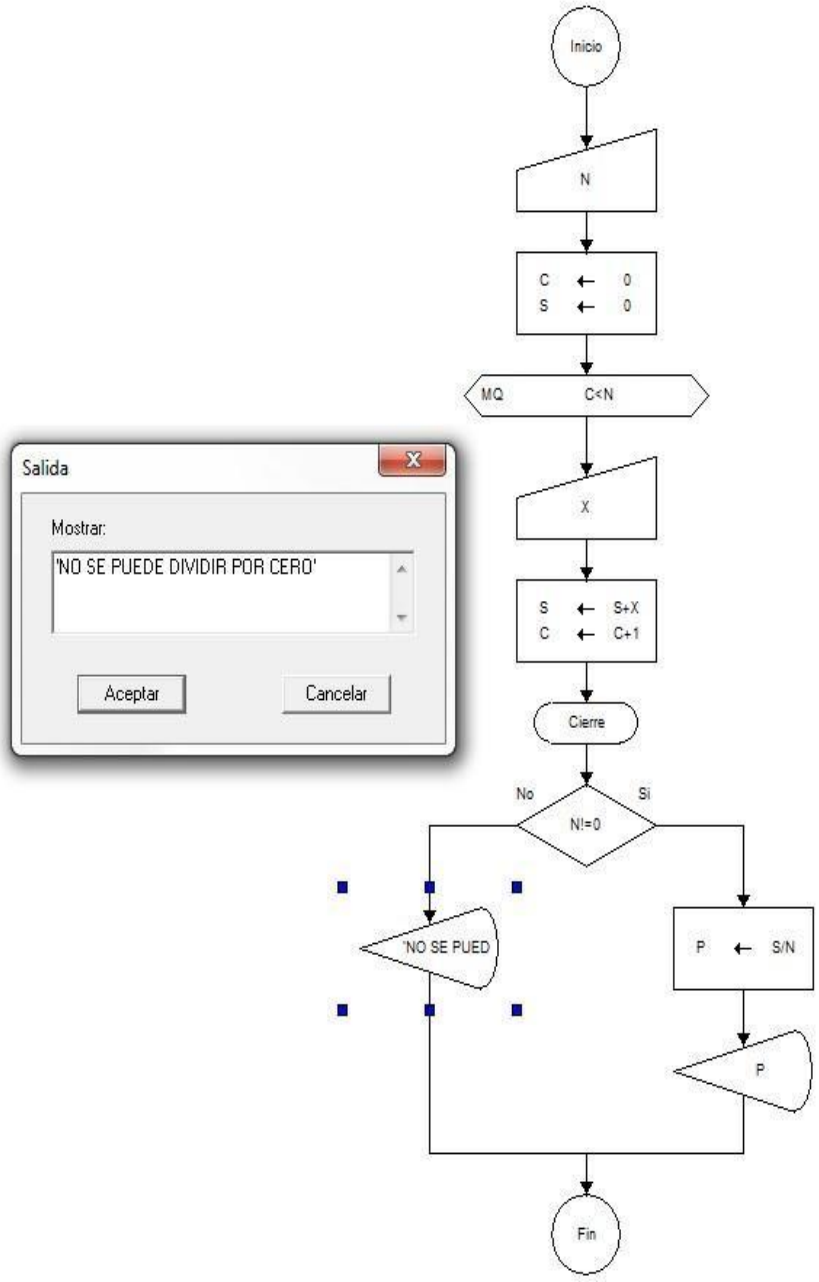
Condiciones Vinculantes

Son aquellas acciones necesarias de realizar con los datos que brinda el problema para conseguir los resultados esperados.

Algunas de estas operaciones pueden ser evaluaciones o decisiones.

En nuestro caso particular, las condiciones vinculantes son dos:

1) realizar la suma de los números ingresados,
2) realizar el cociente entre la suma y N para obtener el promedio aritmético, previa verificación que N sea distinto de 0 (cero), ya que en este caso se produciría un error de tipo lógico.



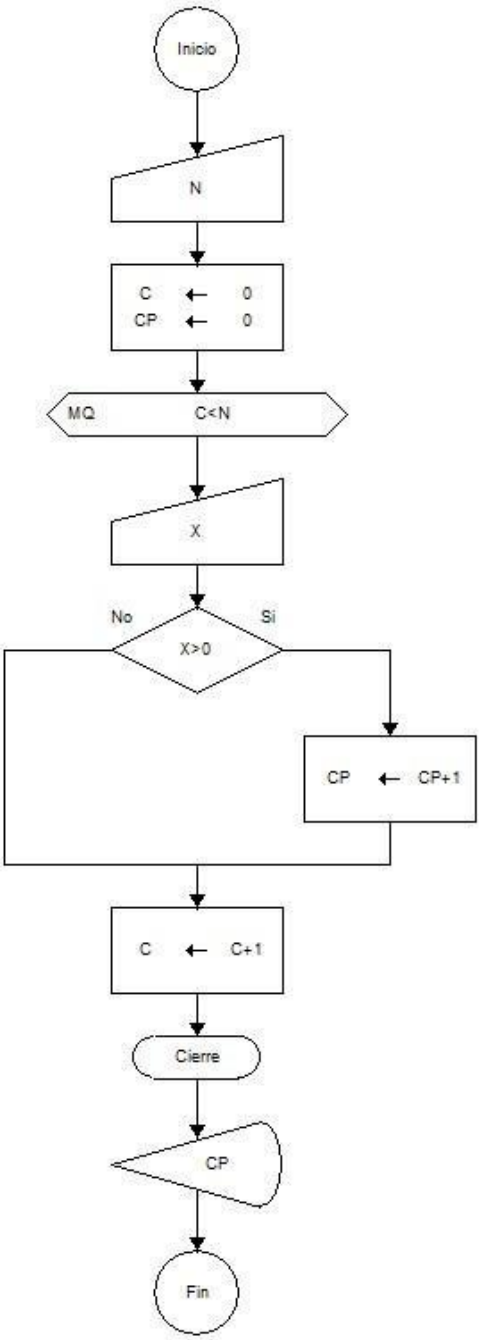
Ejemplo 4:
Introduzca N números enteros y muestre la cantidad de números positivos ingresados.

Datos
N: identificador que indica la cantidad de números a ingresar. X: identificador que indica a cada uno de los números.

Resultado

CP: identificador que indica la cantidad de números positivos introducidos.

En este caso se solicita determinar la **cantidad** de números positivos, por lo tanto, es necesario introducir una variable contadora para este propósito.
Esta variable se incrementará en una unidad cada vez que se ingrese un número positivo.



Condiciones Vinculantes

Condición lógica para saber si un número N es positivo: N>0

Ejemplo 5

Introduzca N números enteros y muestre el porcentaje de números positivos ingresados.

Datos

N: identificador que indica la cantidad de números a ingresar.
X: identificador que indica a cada uno de los números.

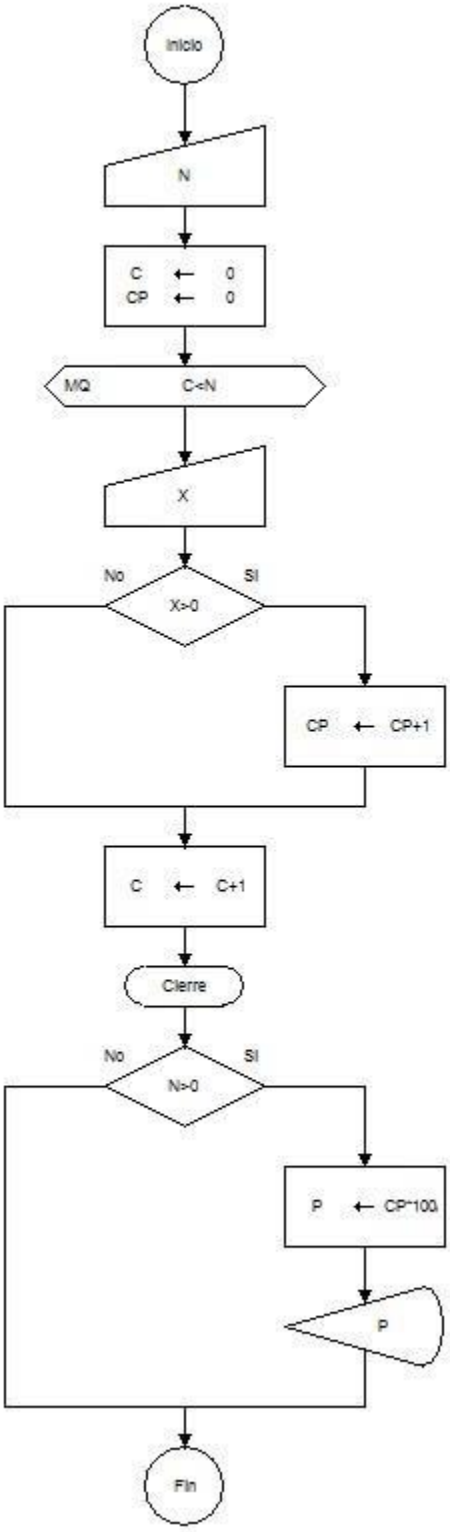
Resultado

P: identificador que indica el porcentaje de números positivos introducidos.

Condiciones Vinculantes

Para calcular porcentajes siempre este debe ser con respecto a algo, en este caso nos piden sobre los números positivos, para la cual se debe **contar** la cantidad de ellos; para tal fin se debe introducir un contador.
La fórmula del porcentaje se consigue con una regla de tres simple en donde se considera que:
Cantidad Total de Valores **N** es el 100%.
El porcentaje solicitado se calcula con respecto al valor de la variable **contador**.

Ejemplo: P=CP*100/N



Ejemplo 6

Introduzca N valores **enteros** y **positivos**, determine el mayor de ellos.

Datos

N: identificador que indica la cantidad de números a ingresar.
X: identificador que indica a cada uno de los números.

Resultado

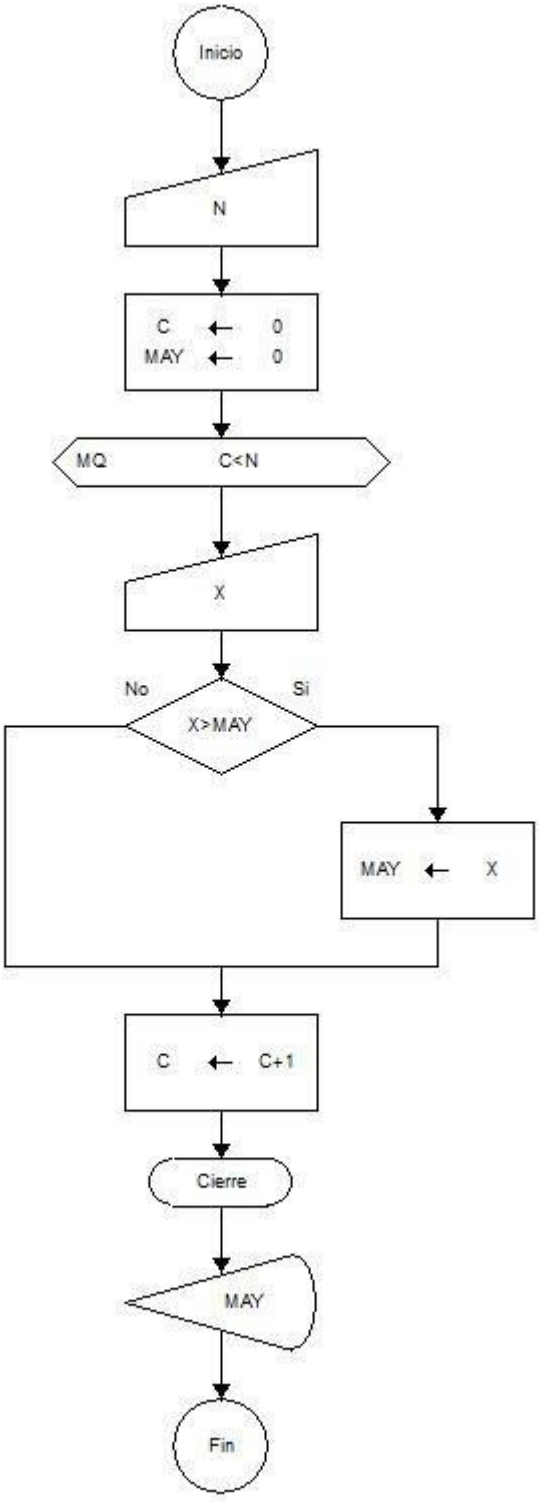
MAY: identificador que indica el mayor valor.

Condiciones Vinculantes

Para poder encontrar el mayor valor se puede realizar de dos formas:

Como en este caso se conoce el rango de los números, se sabe que son enteros y positivos, se puede asignar a la variable mayor el menor valor posible dentro del rango de esos números (0 cero) ya que cualquier valor posterior será mayor al valor asignado inicialmente.

De igual forma se puede encontrar el menor de los N valores ingresados.



Ejemplo 7

Introduzca N valores y determine el mayor de ellos.

En este caso no conocemos el rango de valores de los números, ya que se pueden introducir todos números reales.

Datos
N: identificador que indica la cantidad de números a Ingrese.
X: identificador que indica a cada uno de los números.

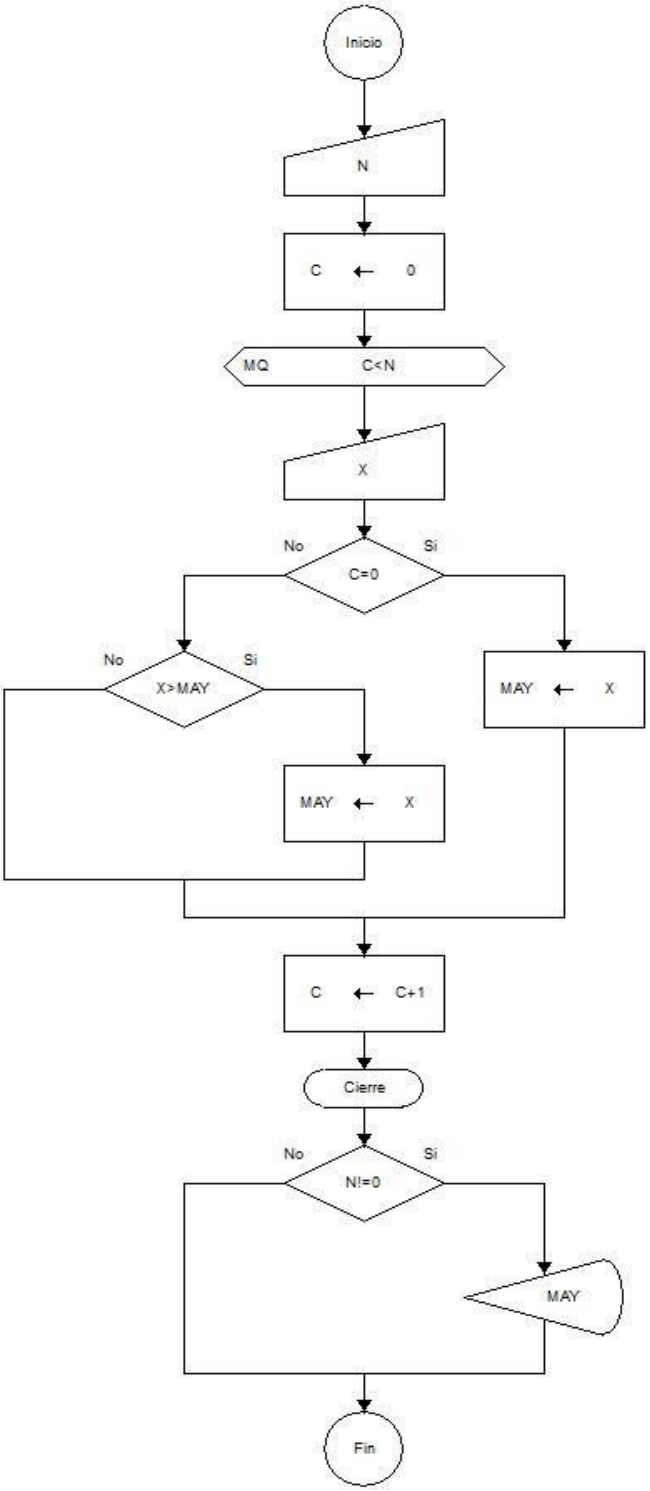
Resultado:
MAY: identificador que indica el mayor valor

Condiciones Vinculantes

En este caso se asigna el primer valor ingresado a la variable mayor (Condición cuando C es igual al valor inicial).

A partir de ella se puedan comenzar a realizar las comparaciones:

Comparaciones del valor MAY con cada uno de los números ingresados.



Ejemplo 8
Introduzca N valores y determine el mayor de ellos y en qué posición se encuentra.

En este caso, lo único que hay que agregar al diagrama anterior es una variable que indique en qué posición ingresó el mayor valor, lo cual está indicado por la variable contadora C.

Datos

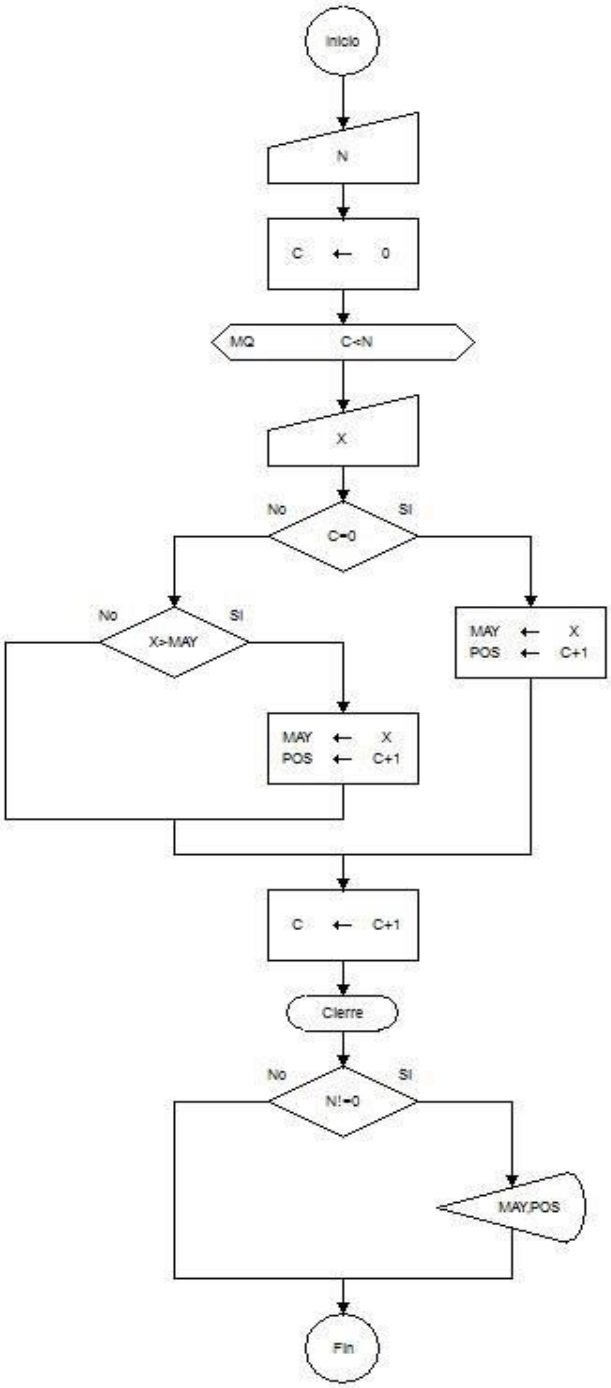
N: identificador que indica la cantidad de números a Ingrese.
X: identificador que indica a cada uno de los números.

Resultado

MAY: identificador que indica el mayor valor
POS: posición en que ingresó el mayor valor

Condiciones Vinculantes

Comparaciones del valor MAY con cada uno de los números ingresados, conservando el mayor valor y la posición en que ingresa.



2. PROBLEMAS DONDE NO SE CONOCE LA CANTIDAD DE VECES QUE SE REPITE EL CICLO.

Ejemplo 1:

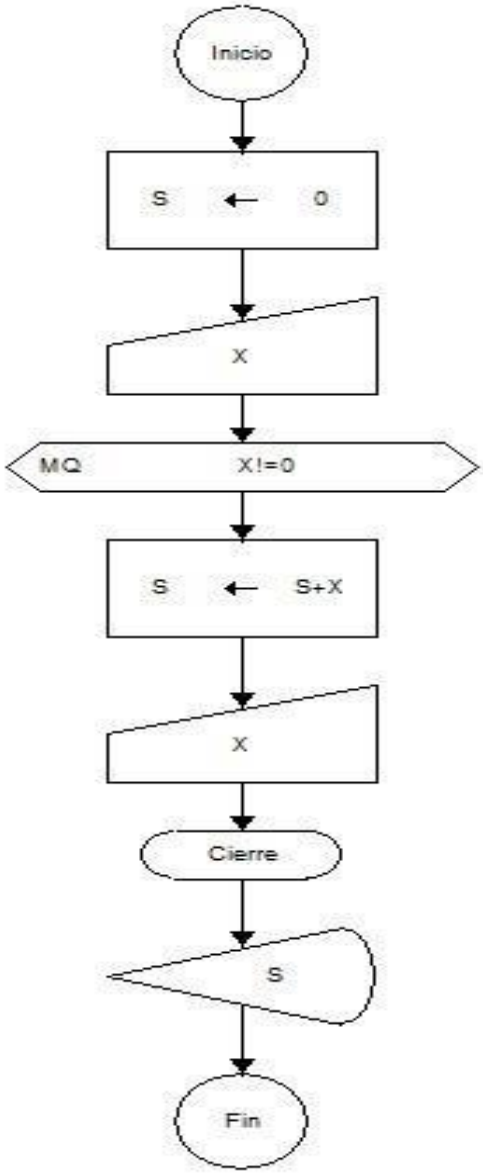
Introduzca una cantidad no determinada de valores, cuyo final está indicado por el valor cero, y determine la suma de ellos.

En este caso los valores se introducen, pero al no conocerse la cantidad de valores que se deben ingresar, no se puede poner un contador para que gobierne el corte de control del ciclo de iteración ya que no tenemos N (que era la cantidad de valores) con quien comparar; por lo cual este corte de control es gobernado por una condición de negación a la dada en el ciclo mientras se utiliza una estructura While.
En este ejemplo la condición de corte será: X!=0.

Dato
X: identificador que indica a cada uno de los números.

Resultado
S: Suma de los valores ingresados

Condiciones Vinculantes
Fórmula del acumulador, para incrementar el valor ingresado.



3. PROBLEMAS DE GENERACIÓN DE VALORES.

Ejemplo 1:
Genere los números impares menores a 100 y dé a conocer la suma de ellos.

En este caso los valores no se introducen, se generan de acuerdo con una determinada fórmula que se da en el enunciado del problema o que es ya conocida.
En este caso se sabe que se comienza con el número 1 y que los números impares se generan sumándole el valor 2 (dos) al número anterior.

Además, los valores a generar deben ser menores a 100.por lo que se puede poner un contador para que gobierne el corte de control del ciclo de iteración. En este ejemplo la condición de corte será: C<=100.

Dato

No se poseen datos pues no se ingresa ninguna variable, todos los valores deben ser generados.

Resultado

S: Suma de los valores generados.

Condiciones Vinculantes

Se comienza con el contador C=1 y se lo incrementa de a dos.