

## Algoritmos

### 1. Introducción

La computadora no solamente es una máquina que puede realizar procesos para darnos resultados, sin que tengamos la noción exacta de las operaciones que realiza para llegar a esos resultados. Con la computadora además de lo anterior también podemos diseñar soluciones a la medida, de problemas específicos que se nos presenten. Más aun, si estos involucran operaciones matemáticas complejas y/o repetitivas, o requieren del manejo de un volumen muy grande de datos.

**Problema:** discrepancia entre un estado inicial y un estado final o deseado. Para pasar de un estado a otro (o sea para resolver el problema) realizamos determinadas tareas o acciones.

Estado inicial → Acciones → Estado final

**Datos:** antecedentes necesarios para llegar al conocimiento exacto de una cosa o para deducir las consecuencias legítimas de un hecho. Pueden ser: numéricos, alfabéticos, signos, fechas, etc.

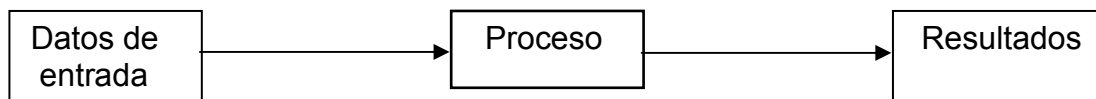
Un dato es una representación de un objeto del mundo real mediante la cual podemos modelar aspectos del problema que se quiere resolver.

**Computadora:** Es un dispositivo electrónico utilizado para procesar información y obtener resultados. Los datos se pueden introducir en la computadora como entrada (input) y a continuación se procesan para producir información de salida (output).

### **Proceso de información en la computadora**

Computadora: es una máquina digital y sincrónica, con cierta capacidad de cálculo numérico y lógico controlado por un programa almacenado y con posibilidad de comunicación con el mundo exterior.

- Su finalidad es ayudar al hombre a realizar *tareas repetitivas en menor tiempo y con mayor exactitud*.
- *No razona ni crea soluciones*, sino que *ejecuta una serie de órdenes* que le proporciona el ser humano.
- Es una máquina capaz de aceptar datos de entrada, ejecutar con ellos cálculos aritméticos y lógicos y dar información de salida (resultados), *bajo control de un programa previamente almacenado en su memoria*.



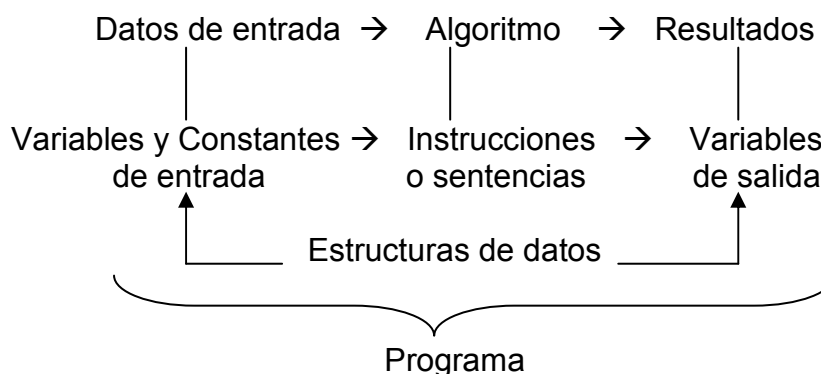
El diseño de soluciones a la medida de nuestros problemas, requiere como en otras disciplinas una metodología que nos enseñe de manera gradual, la forma de llegar a estas soluciones.

### 2. Definición de Algoritmo

La palabra algoritmo se deriva de la traducción al latín de la palabra árabe Alkhwarizmi, nombre de un matemático y astrónomo árabe que escribió un tratado sobre manipulación de números y ecuaciones en el siglo IX.

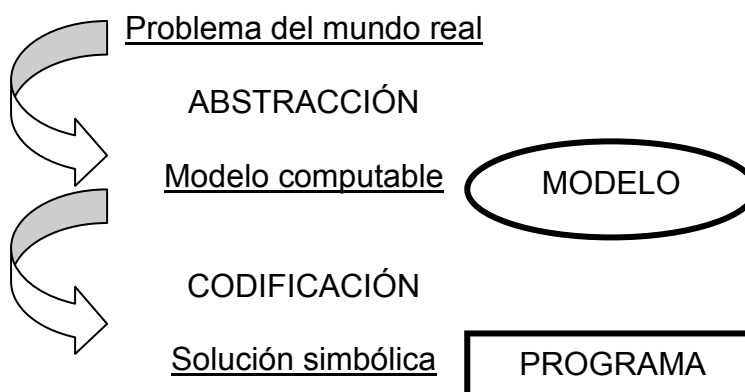
Un algoritmo es una serie de pasos organizados que describe el proceso que se debe seguir, para dar solución a una clase de problemas. Por ejemplo el procedimiento para resolver manualmente una multiplicación entre dos números.

**Algoritmo:** es un método para resolver una clase de problemas, que indicará exactamente como se obtendrán los resultados deseados. Cada una de las acciones a realizar se denomina **instrucción** (o sentencia, o primitiva). Las instrucciones han de indicarse en forma ordenada y no ambigua y la cantidad de instrucciones ha de ser finita. El mismo problema puede ser resuelto por más de un algoritmo.



A las soluciones creadas por computadora se les conoce como **programas** y no son más que una serie de operaciones que realiza la computadora para llegar a un resultado, con un grupo de datos específicos.

**Programa:** Es el conjunto de instrucciones escritas de algún lenguaje de programación y que ejecutadas secuencialmente resuelven un problema. Un programa es un algoritmo cuyas instrucciones pueden ser ejecutadas por una computadora.



- El proceso de análisis del mundo real para interpretar los aspectos esenciales de un problema y ponerlo en términos precisos se denomina **abstracción**.
- Abstraer un problema del mundo real y simplificar su expresión, tratando de encontrar los aspectos principales que se pueden resolver (requerimientos), los datos que se han de procesar y el contexto del problema se denomina **modelización**.

Para poder realizar *programas*, además de conocer la metodología mencionada, también debemos de conocer, de manera específica las funciones que pueden realizar la computadora y las formas en que se pueden manejar los elementos que hay en la misma.

### 3. Tipos De Datos

Todos los datos tienen un tipo asociado con ellos. Un dato puede ser un simple **carácter**, tal como 'b', un valor entero tal como 35 o un valor real como 35.24. El tipo de dato determina la naturaleza del conjunto de valores que puede tomar una variable y las operaciones que puede realizar sobre ellos. Veremos más adelante que existen otros tipos de datos estructurados formados por combinación de estos tipos simples.

### 4. Constantes y Variables

**Constante:** Una constante es un dato numérico o alfanumérico que no cambia durante la ejecución del programa.

Ejemplo:

pi = 3.1416

**Variable:** Es un espacio en la memoria de la computadora que permite almacenar temporalmente un dato durante la ejecución de un proceso, su contenido puede cambiar durante la ejecución del programa. Para poder reconocer una variable en la memoria de la computadora, es necesario darle un nombre con el cual podamos identificarla dentro de un algoritmo.

Ejemplo:

area = pi \* radio \* radio

Las variables son: el radio, el área y la constante es pi

El símbolo = se emplea para asignar un valor a una variable.

### 5. Expresiones

Las expresiones son combinaciones de constantes, variables, símbolos de operación, paréntesis y nombres de funciones especiales. Por ejemplo:

$a + (b + 3) / c$

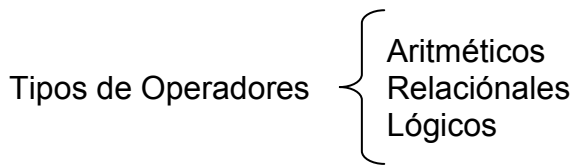
Cada expresión toma un valor que se determina tomando los valores de las variables y constantes implicadas y la ejecución de las operaciones indicadas.

Una expresión consta de operadores y operandos. Según sea el tipo de datos que manipulan, se clasifican las expresiones en:

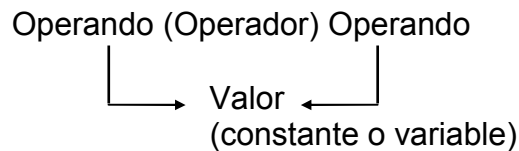
- Aritméticas
- Relacionales
- Lógicas

### 6. Operadores y Operandos

**Operadores:** Son elementos que relacionan de forma diferente, los valores de una o mas variables y/o constantes. Es decir, los operadores nos permiten manipular valores.



**Operadores Aritméticos:** Los operadores aritméticos permiten la realización de operaciones matemáticas con los valores (variables y constantes).



### Operadores Aritméticos

+	Suma
-	Resta
*	Multiplicación
/	División
%	Resto (residuo de la división entera)

Ejemplos:

Expresión	Resultado
7 / 2	3.5
12 % 7	5
4 + 2 * 5	14

### Prioridad de los Operadores Aritméticos

- Todas las expresiones entre paréntesis se evalúan primero. Las expresiones con paréntesis anidados se evalúan de dentro a fuera, el paréntesis más interno se evalúa primero.
- Dentro de una misma expresión los operadores se evalúan en el siguiente orden.
  - 2.- \*, /, % Multiplicación, división, resto.
  - 3.- +, - Suma y resta.
- Los operadores en una misma expresión con igual nivel de prioridad se evalúan de izquierda a derecha.

Ejemplos:

$$\begin{array}{ll}
 4 + 2 * 5 = 14 & 46 / 5 = 9.2 \\
 23 * 2 / 5 = 9.2 & 3 + 5 * (10 - 6) = 3 + 5 * 4 = 3 + 20 = 23 \\
 3 + 5 * (10 - (2 + 4)) = 23 & 3.5 + 5.09 - 14.0 / 40 = 5.09 \\
 3.5 + 5.09 - 14.0 / 40 = 5.09 & 2.1 * (1.5 + 3.0 * 4.1) = 28.98 \\
 2.1 * (1.5 + 3.0 * 4.1) = 28.98 & 2.1 * (1.5 + 12.3) = 2.1 * 13.8 = 28.98
 \end{array}$$

### Operadores Relacionales:

- Se utilizan para establecer una relación entre dos valores.
- Compara estos valores entre si y esta comparación produce un resultado de certeza o falsedad (verdadero o falso).

- Los operadores relacionales comparan valores del mismo tipo.
- Tienen el mismo nivel de prioridad en su evaluación.
- Los operadores relacionales tiene menor prioridad que los aritméticos.

### Operadores Relacionales

>	Mayor que
<	Menor que
> =	Mayor o igual que
< =	Menor o igual que
!=	Diferente
==	Igual

Ejemplos:

Si  $a = 10$      $b = 20$      $c = 30$

$a + b > c$	Falso
$a - b < c$	Verdadero
$a - b == c$	Falso
$a * b != c$	Verdadero

Ejemplos no lógicos:

$a < b < c$   
 $10 < 20 < 30$   
 $T < 30$     (T = True = verdadero)

No es correcto porque tiene diferentes tipos de operandos.

### Operadores Lógicos:

- Estos operadores se utilizan para establecer relaciones entre valores lógicos.
- Estos valores pueden ser resultado de una expresión relacional.

#### Operadores Lógicos

&&	Y (and)
	O (or)
!	Negación

#### Tabla de verdad del Operador &&

Operando1	Operador	Operando2	Resultado
F	&&	F	F
F	&&	T	F
T	&&	F	F
T	&&	T	T

#### Operador ||

Operando1	Operador	Operando2	Resultado
F		F	F
F		V	V
V		F	V
V		V	V

Note que esta última fila de la tabla difiere del O que se utiliza en el lenguaje corriente.



### **Reglas para formar un Identificador**

- Debe comenzar con una letra (A a Z, mayúsculas o minúsculas) y no deben contener espacios en blanco.
- Letras, dígitos y caracteres como la subraya ( \_ ) están permitidos después del primer carácter.

## **8. Clasificación de las Variables por su Uso**

**Variables de Trabajo:** Variables que reciben el resultado de una operación matemática completa y que se usan normalmente dentro de un programa. Ejemplo:  
`suma=a+b/c`

**Contadores:** Se utilizan para llevar el control del número de ocasiones en que se realiza una operación o se cumple una condición. Con los incrementos generalmente de uno en uno.

Tienen la forma:

`variable = variable + incremento`  
`variable = variable – decremento`

Note que **variable** a la derecha del = (asignación) tiene el valor actual y **variable** a la izquierda del igual tendrá el nuevo valor.

Ejemplo:

`A = A + 1`

**Acumuladores:** Forma que toma una variable y que sirve para llevar la suma acumulativa de una serie de valores que se van leyendo o calculando progresivamente.

Tienen la forma:

`variable1 = variable1 + variable2`

**Banderas:** Se utilizan para memorizar una condición lógica.

## **9. Funciones internas**

Las operaciones que se requieren en los programas exigen en numerosas ocasiones, además de las operaciones aritméticas básicas, un número determinado de operadores especiales que se denominan funciones internas, incorporadas o estándar. A continuación, se muestra una tabla con algunas funciones internas que se encuentran en la mayoría de los lenguajes de programación, se toma x como argumento de la función.

Función	Tipo de argumento	Resultado
Valor absoluto de x	Entero o real	Igual a argumento

Arco tangente de x	Entero o real	Real
Coseno de x	Entero o real	Real
Exponencial de x	Entero o real	Real
Logaritmo neperiano de x	Entero o real > 0	Real
Seno de x	Entero o real	Real
Cuadrado de x	Entero o real	Igual a argumento
Truncamiento de x	Real	Entero
Raíz cuadrada de x	Entero o real >=0	Real
Logaritmo decimal de x	Entero o real	Real
Potencia x elevado a la y	Entero o real	Real

## 10. Metodología para la solución de problemas por medio de computadora

### Definición del Problema

Esta fase está dada por el enunciado del problema, el cual requiere una definición clara y precisa. Es importante que se conozca lo que se desea que realice la computadora; mientras esto no se conozca del todo no tiene mucho caso continuar con la siguiente etapa.

### Análisis del Problema

Una vez que se ha comprendido lo que se desea de la computadora, es necesario definir:

Los datos de entrada.

Cual es la información que se desea producir (salida)

Los métodos y fórmulas que se necesitan para procesar los datos.

Una recomendación muy practica es el que nos pongamos en el lugar de la computadora y analicemos que es lo que necesitamos que nos ordenen y en que secuencia para producir los resultados esperados.

### Diseño del Algoritmo

Las características de un buen algoritmo son:

Debe tener un punto particular de inicio.

Debe ser definido, no debe permitir dobles interpretaciones.

Debe ser general, es decir, soportar la mayoría de las variantes que se puedan presentar en la definición del problema.

Debe ser finito en tamaño y tiempo de ejecución.

### Codificación

La codificación es la operación de escribir la solución del problema (de acuerdo a la lógica del diagrama de flujo), en una serie de instrucciones detalladas, en un código reconocible por la computadora, la serie de instrucciones detalladas se le conoce como código fuente, el cual se escribe en un lenguaje de programación o lenguaje de alto nivel.

### Prueba y Depuración

Los errores humanos dentro de la programación de computadoras son muchos y aumentan considerablemente con la complejidad del problema. El proceso de identificar y eliminar errores, para dar paso a una solución sin errores se le llama **depuración**.



La **depuración o prueba** resulta una tarea tan creativa como el mismo desarrollo de la solución, por ello se debe considerar con el mismo interés y entusiasmo. Resulta conveniente observar los siguientes principios al realizar una depuración, ya que de este trabajo depende el éxito de nuestra solución.

### **Documentación**

Es la guía o comunicación escrita en sus variadas formas, ya sea en enunciados, procedimientos, dibujos o diagramas.

A menudo un programa escrito por una persona, es usado por otra. Por ello la documentación sirve para ayudar a comprender o usar un programa o para facilitar futuras modificaciones (mantenimiento).

La **documentación** se divide en tres partes:

Documentación Interna

Documentación Externa

Manual del Usuario

- Documentación Interna: Son los comentarios o mensaje que se añaden al código fuente para hacer mas claro el entendimiento de un proceso.
- Documentación Externa: Se define en un documento escrito los siguientes puntos:
  - Descripción del Problema
  - Nombre del Autor
  - Algoritmo (diagrama de flujo o pseudocódigo)
  - Diccionario de Datos
  - Código Fuente (programa)
- Manual del Usuario: Describe paso a paso la manera como funciona el programa, con el fin de que el usuario obtenga el resultado deseado.

### **Mantenimiento**

Se lleva a cabo después de terminado el programa, cuando se detecta que es necesario hacer algún cambio, ajuste o complementación al programa para que siga trabajando de manera correcta. Para poder realizar este trabajo se requiere que el programa este correctamente documentado.

## **11. Técnicas para la formulación de algoritmos**

### **Lenguajes Algorítmicos**

Es una serie de símbolos y reglas que se utilizan para describir de manera explícita un proceso.

### **Tipos de Lenguajes Algorítmicos**


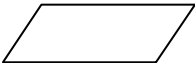

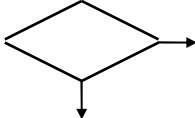


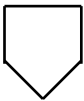
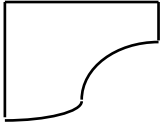

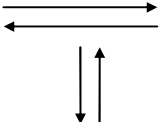
- **Gráficos**: Es la representación gráfica de las operaciones que realiza un algoritmo (diagrama de flujo).
- **No Gráficos**: Representa en forma descriptiva las operaciones que debe realizar un algoritmo (pseudocódigo).

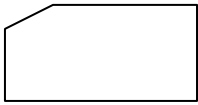
## Diagrama de Flujo

Un diagrama de flujo es la representación gráfica de un algoritmo. También se puede decir que es la representación detallada en forma gráfica de como deben realizarse los pasos en la computadora para producir resultados.

Esta representación gráfica se da cuando varios símbolos (que indican diferentes procesos en la computadora), se relacionan entre si mediante líneas que indican el orden en que se deben ejecutar los procesos.

Los símbolos utilizados han sido normalizados por el instituto norteamericano de normalización (ANSI).

<b><u>SÍMBOLO</u></b>	<b><u>DESCRIPCIÓN</u></b>
	Indica el inicio y el final de nuestro diagrama de flujo.
	Indica la entrada y salida de datos.
	Símbolo de proceso y nos indica la asignación de un valor en la memoria y/o la ejecución de una operación aritmética.
	Símbolo de decisión indica la realización de una comparación de valores.
	Se utiliza para representar los subprogramas.
	Conector dentro de página. Representa la continuidad del diagrama dentro de la misma página.
	Conector fuera de página. Representa la continuidad del diagrama en otra página.
	Indica la salida de información por impresora.
	Indica la salida de información en la pantalla o monitor.
	Líneas de flujo o dirección. Indican la secuencia en que se realizan las operaciones.



Entrada de datos (el grafico representa las históricas tarjetas Perforadas)

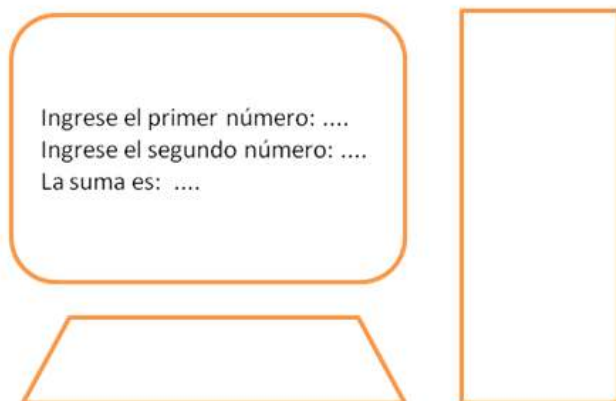
### **Recomendaciones para el diseño de Diagramas de Flujo**

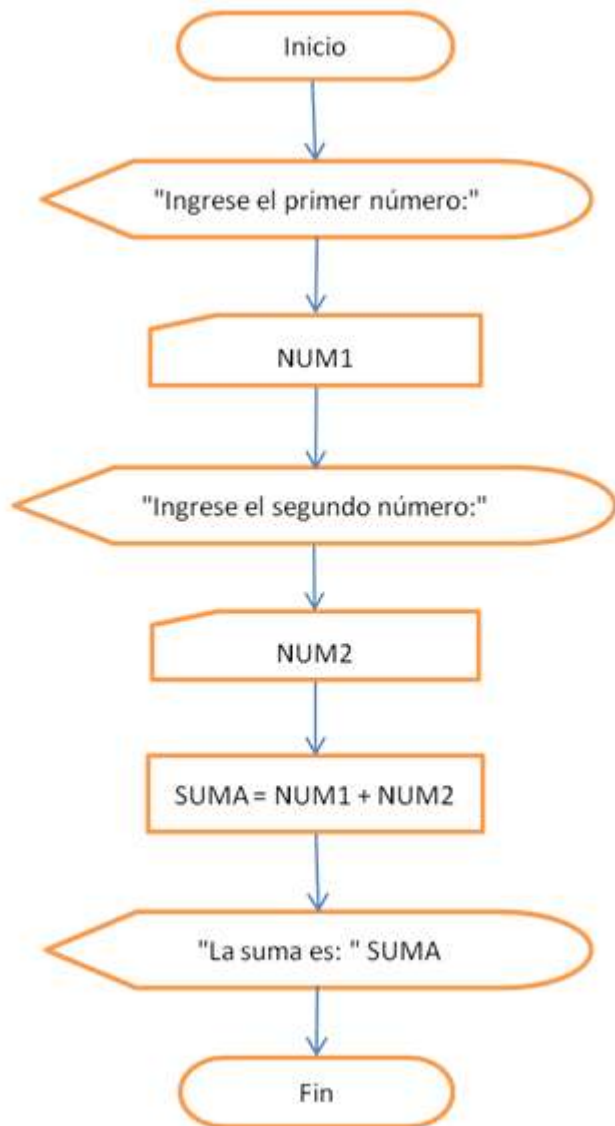
- Se deben usar solamente líneas de flujos horizontales y/o verticales.
- Se debe evitar el cruce de líneas utilizando los conectores.
- Se deben usar conectores solo cuando sea necesario.
- No deben quedar líneas de flujo sin conectar.
- Se deben trazar los símbolos de manera que se puedan leer de arriba hacia abajo y de izquierda a derecha.
- Todo texto escrito dentro de un símbolo deberá ser escrito claramente, evitando el uso de muchas palabras.
- Cuando el nombre de una variable no es autoexplicativo, debe aclararse que dato contiene.

Ejemplos simples:

#### 1) Algoritmo secuencial:

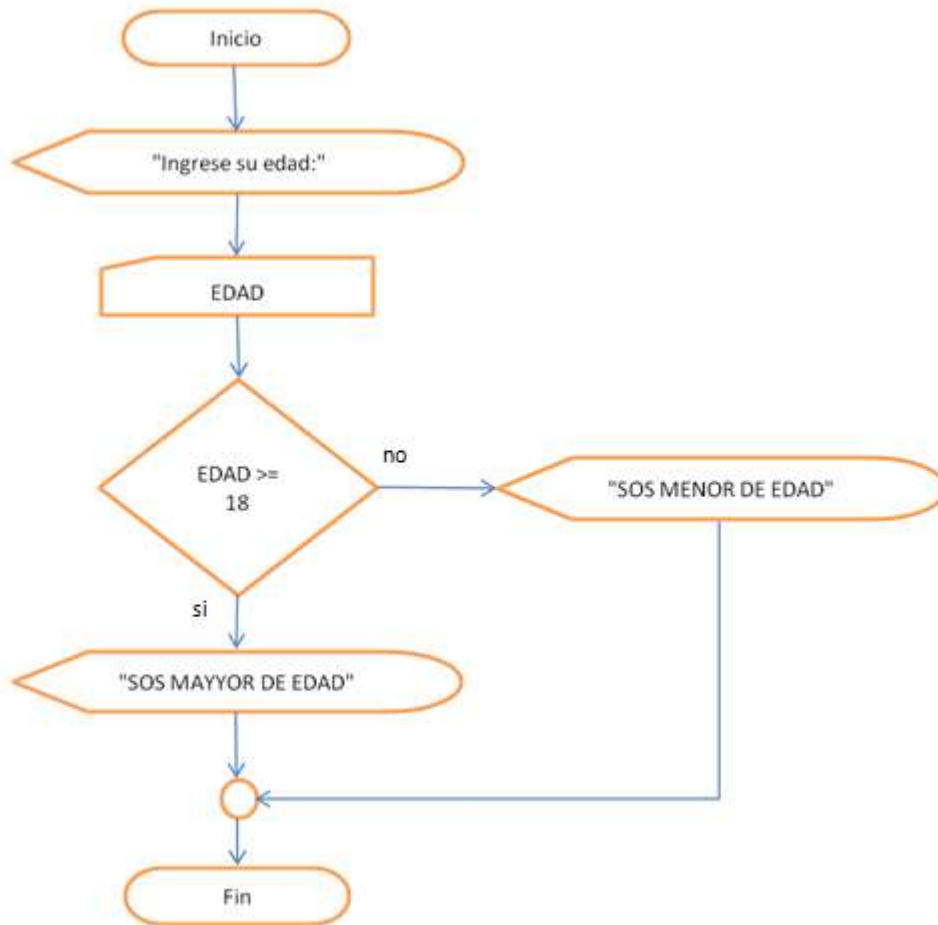
Realizar un diagrama de flujo que permita resolver el siguiente problema: se ingresan dos números, mostrar la suma de los mismos:





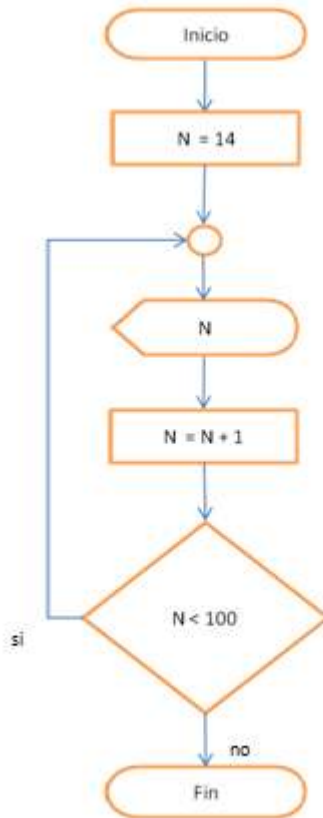
## 2) Algoritmo condicional:

Realizar un diagrama de flujo que permita resolver el siguiente problema: se ingresa la edad de una persona, determinar si es mayor o menor de edad.



### 3) Algoritmo iterativo o repetitivo:

Realizar un diagrama de flujo que permita resolver el siguiente problema: mostrar en pantalla los múltiplos de 7 menores que 100.



4) Tipos de variables según su función (ver arriba ítem 8):

Realizar un diagrama de flujo que permita resolver el siguiente problema: Se ingresan las notas de un parcial de los alumnos de un curso y se debe calcular el promedio general del curso. De antemano no se conocen cuantos alumnos rindieron.

