Programación

Diagramas de flujo y pseudocódigo

Programación

Diagramas de flujo

Programación

Diagramas de flujo

Los diagramas de flujo son herramientas gráficas para la representación visual y gráfica de algoritmos, compuestos por una serie de símbolos icónicos unidos por flechas.

Programación

Diagramas de flujo

Características:

Los símbolos representan acciones o funciones en el programa.
Las flechas representan el orden de realización de las acciones o funciones, marcando el sentido o flujo lógico del algoritmo.
Cada símbolo tendrá al menos una flecha que conduzca a él y una flecha que parta de él, a excepción de los terminadores y conectores.
Se leen de arriba a abajo y de izquierda a derecha.

Programación

Diagramas de flujo

Los diagramas de flujo son herramientas gráficas para la representación visual y gráfica de algoritmos, compuestos por una serie de símbolos icónicos unidos por flechas.

Ventajas:

- Al ser visuales son muy sencillos de entender.
- Utilizan símbolos estándar.

Desventaja

- La dificultad de mantenimiento y actualización, puesto que hay que utilizar editores gráficos.

Pseudocódigo

Programación

Pseudocódigo

El pseudocódigo es un pseudolenguaje intermedio entre el natural del programador y el lenguaje de programación seleccionado, considerándose por tanto un lenguaje de pseudoprogramación.

Programación

Pseudocódigo

El pseudocódigo es un pseudolenguaje intermedio entre el natural del programador y el lenguaje de programación seleccionado, considerándose por tanto un lenguaje de pseudoprogramación.

No existe una sintaxis estándar para el pseudocódigo, utilizando una mezcla de lenguaje natural (utilizando como base la lengua nativa del programador) y una serie de símbolos, términos y otras características propias de los lenguajes de programación de alto nivel como Pascal o C.

Programación

Pseudocódigo

El pseudocódigo es un pseudolenguaje intermedio entre el natural del programador y el lenguaje de programación seleccionado, considerándose por tanto un lenguaje de pseudoprogramación.

Sus principales características son:

- Es fácil de aprender y utilizar.
- Es conciso.
- Es independiente del lenguaje de programación que se vaya a utilizar.
- Facilita el paso del programa al lenguaje de programación.
- Es fácil de mantener.

Programación

Pseudocódigo

El pseudocódigo es un pseudolenguaje intermedio entre el natural del programador y el lenguaje de programación seleccionado, considerándose por tanto un lenguaje de pseudoprogramación.

Este es un ejemplo de pseudocódigo (para el juego matemático bizz buzz):

Pseudocódigo estilo Fortran:

```
programa bizzbuzz
hacer i = 1 hasta 100
    establecer print_number a verdadero
    si i es divisible por 3
        escribir "Bizz"
        establecer print_number a falso
    si i es divisible por 5
        escribir "Buzz"
        establecer print_number a falso
    si print_number, escribir i
    escribir una nueva línea
fin del hacer
```

Pseudocódigo estilo Pascal:

```
procedimiento bizzbuzz
para i := 1 hasta 100 hacer
    establecer print_number a verdadero;
    Si i es divisible por 3 entonces
        escribir "Bizz";
        establecer print_number a falso;
    Si i es divisible por 5 entonces
        escribir "Buzz";
        establecer print_number a falso;
    Si print_number, escribir i;
    escribir una nueva línea;
fin
```

Pseudocódigo estilo C:

```
subproceso funcion bizzbuzz
para (i <- 1; i<=100; i++) {
    establecer print_number a verdadero;
    Si i es divisible por 3
        escribir "Bizz";
        establecer print_number a falso;
    Si i es divisible por 5
        escribir "Buzz";
        establecer print_number a falso;
    Si print_number, escribir i;
    escribir una nueva línea;
}
```

Programación

Pseudocódigo

El pseudocódigo es un pseudolenguaje intermedio entre el natural del programador y el lenguaje de programación seleccionado, considerándose por tanto un lenguaje de pseudoprogramación.

Sus principales ventajas frente a las técnicas diagramáticas son su facilidad de creación, evolución y mantenimiento, y la facilidad para expresar el pseudocódigo en cualquier lenguaje de programación.

Sus mayores inconvenientes son su falta de estandarización y la dificultad para su lectura cuando su tamaño crece.

Diagramas de flujo
Símbolos

Programación

Diagramas de flujo (Símbolos)

Representación gráfica de los algoritmos

Los algoritmos se representan mediante los diagramas de flujo. Estos utilizan una serie de símbolos (cajas), unidos por líneas de flujo que indican la secuencia en que deben ejecutarse los programas. A estos diagramas también se los llama **organigramas,** ya que señalan el orden de ejecución.

Los símbolos más habituales son los que se muestran a la derecha, y su interpretación es la siguiente:

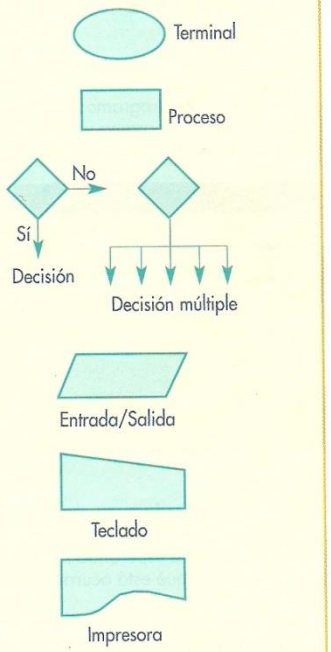
Terminal: se utiliza para representar el inicio y el fin de un programa; también puede representar una pausa o interrupción del programa.

Proceso: representa cualquier tipo de operación que se pueda llevar a cabo con los datos.

Decisión: se usa para representar una operación lógica o una comparación de datos para que, en función del resultado, el programa tome uno de los caminos. Lo normal es que tenga dos salidas, pero puede tener más, en cuyo caso se denomina **decisión múltiple.**

Entrada/Salida: simboliza la introducción de datos o la salida de información a través de cualquier periférico. Este se puede sustituir por símbolos específicos según el periférico utilizado.

Líneas de flujo: indican la secuencia en la que se van a ejecutar los pasos del programa.



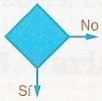
Programación

Diagramas de flujo (Símbolos)

Líneas de flujo: indican la secuencia en la que se van a ejecutar los pasos del algoritmo.

Terminal: se utiliza para representar el inicio y el final de la tarea a realizar. También puede representar una pausa o interrupción.

Proceso: representa una o varias instrucciones que se realizan de forma secuencial.



Decisión: se usa para representar una operación lógica o una comparación de datos para que, en función del resultado, se siga por un camino u otro. Lo normal es que tenga dos salidas, pero puede tener más, en cuyo caso se denomina «decisión múltiple».

Función: es un módulo independiente que realiza una tarea determinada. Permite agrupar varias instrucciones que se realizan de forma repetitiva. Por ejemplo, dibujar un círculo a partir de su radio.

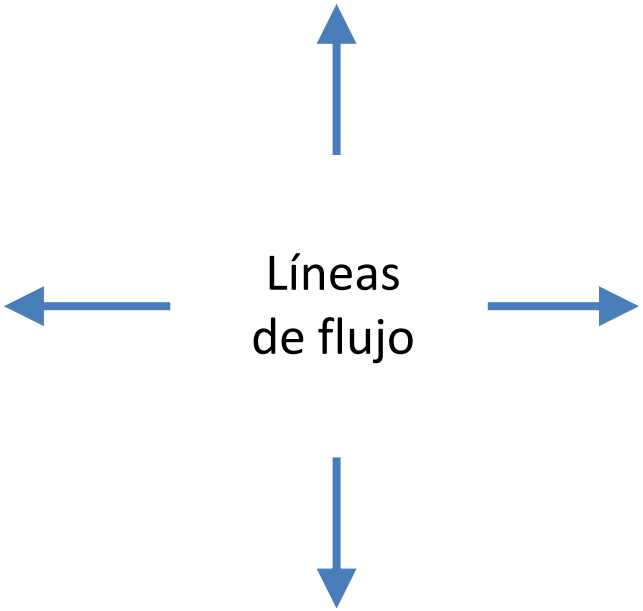
Entrada/Salida: simboliza la introducción de datos o la salida de información a través de cualquier medio. En el caso de un sistema informático se puede sustituir por cualquiera de los siguientes periféricos:

Teclado Pantalla Impresora Disco

Programación

Diagramas de flujo (Símbolos)

Terminales



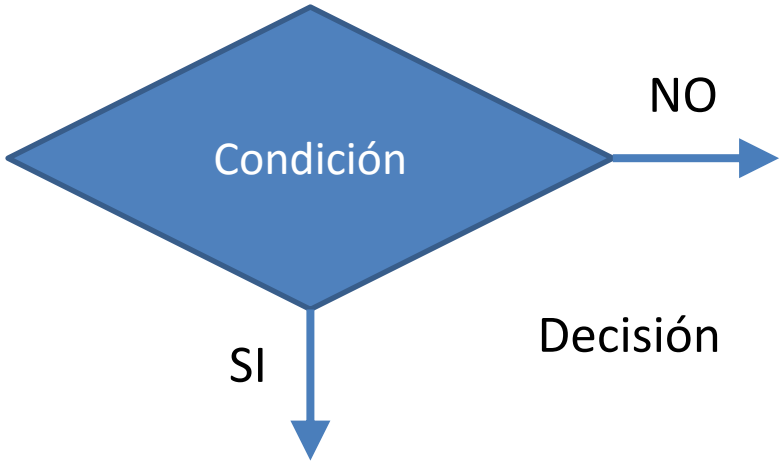
Proceso

Programación

Diagramas de flujo (Símbolos)

Proceso

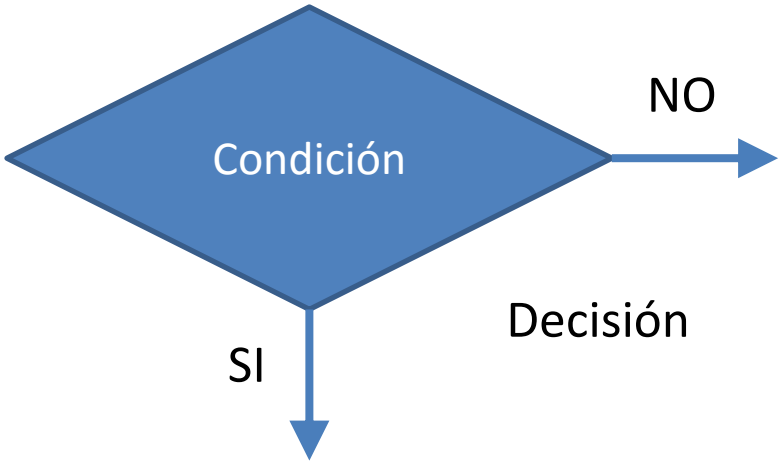
Función



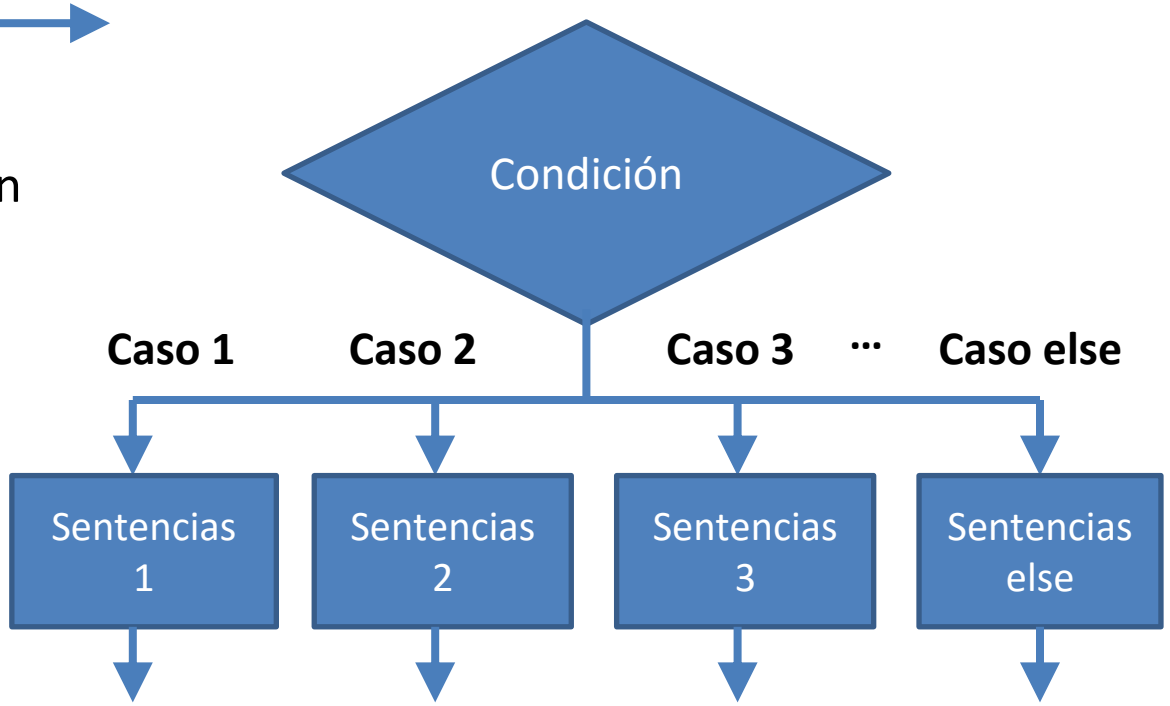
Decisión

Programación

Diagramas de flujo (Símbolos)



Decisión

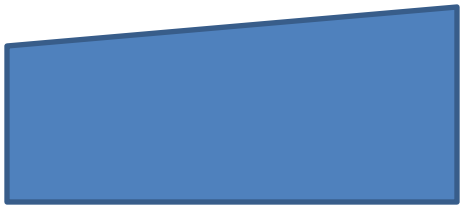


Decisión múltiple

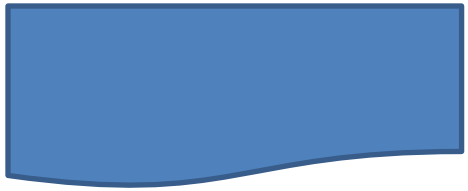
Programación

Diagramas de flujo (Símbolos)

Entrada / Salida

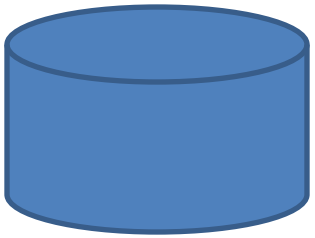


Teclado



Impresora

Pantalla

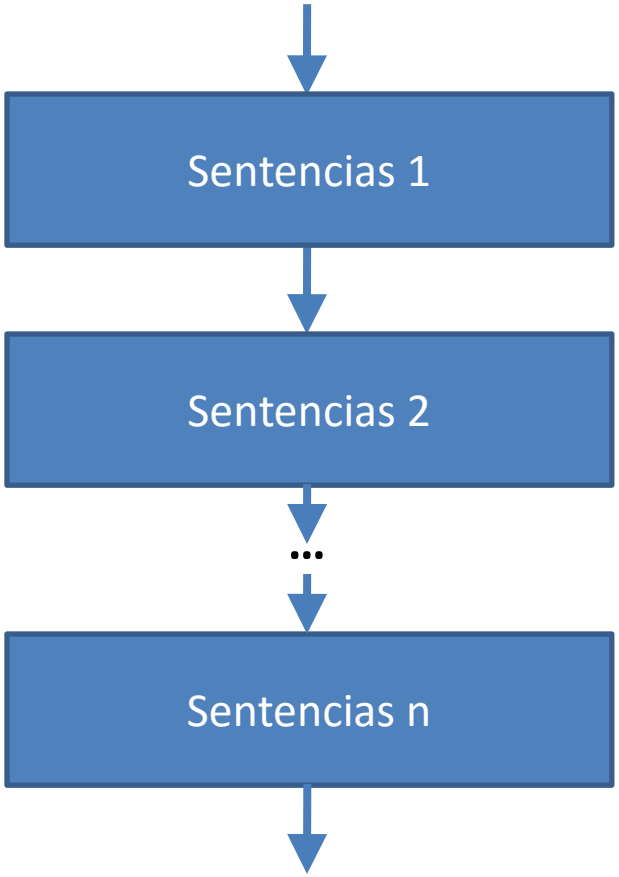


Disco

Estructuras de control de flujo

Programación

Estructura secuencial



Sentencias 1

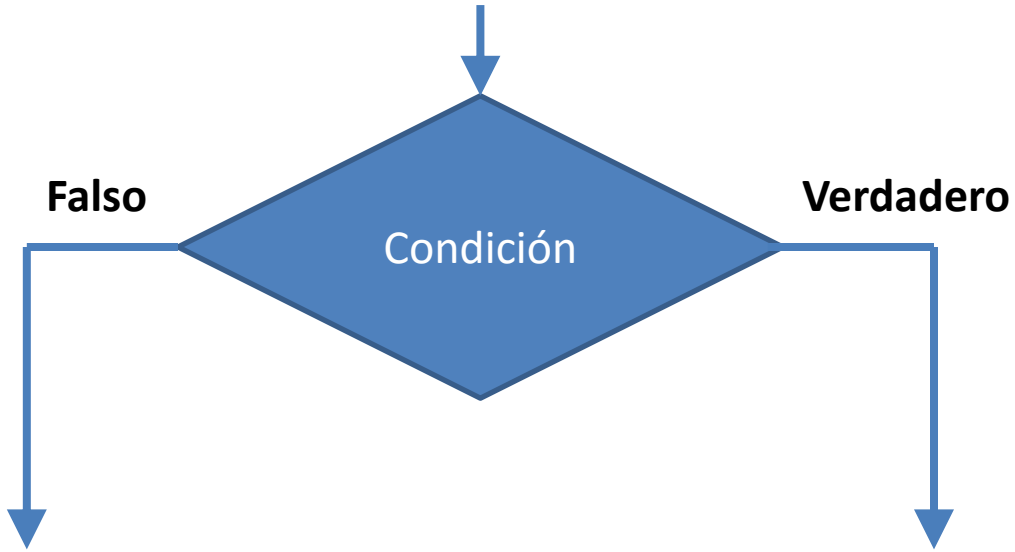
Sentencias 2

...

Sentencias n

Programación

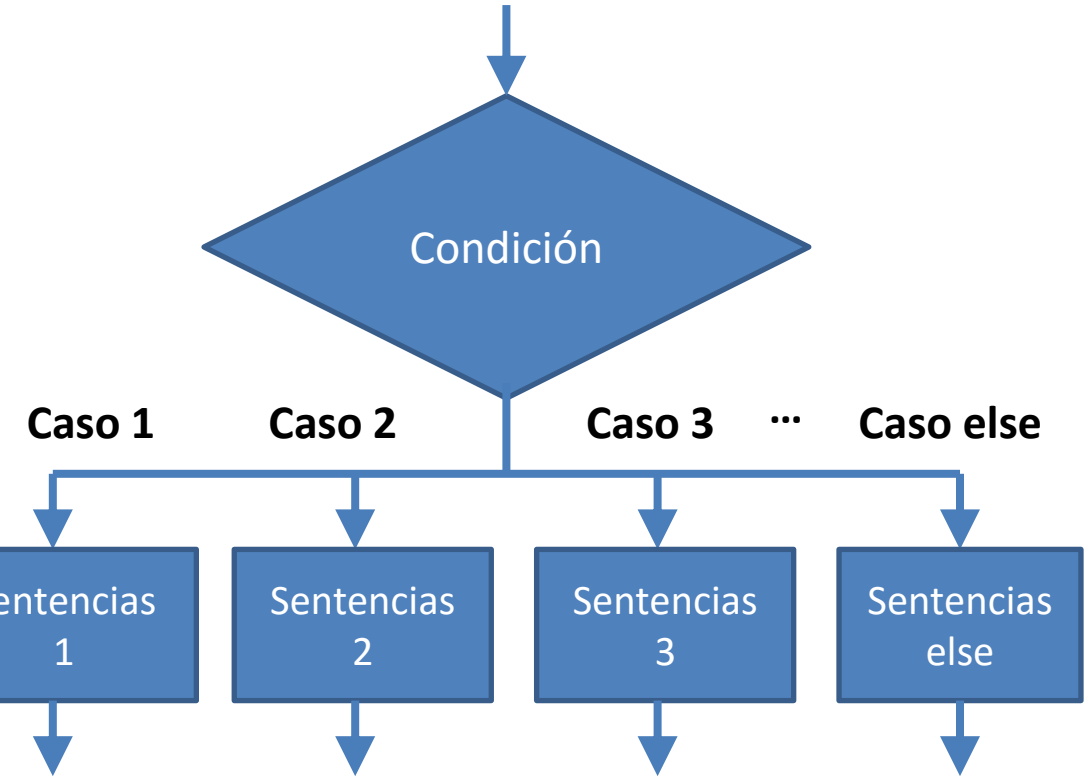
Estructura condicional




```
Si (condición) entonces
    Sentencias 1
else
    Sentencias 2
Fin si
```

Programación

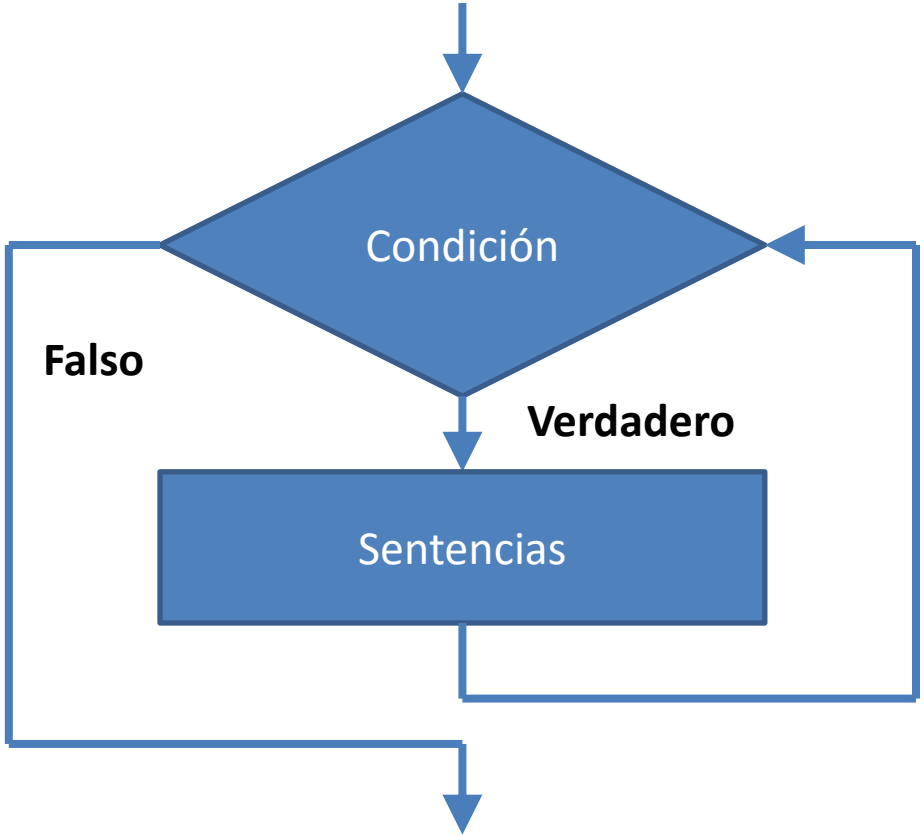
Estructura condicional (Selección múltiple)




```
Si (condición 1) entonces
        Sentencias 1
else si (condición 2) entonces
        Sentencias 2
else si (condición 3) entonces
        Sentencias 3
---
else
        Sentencias else
Fin si
```

Programación

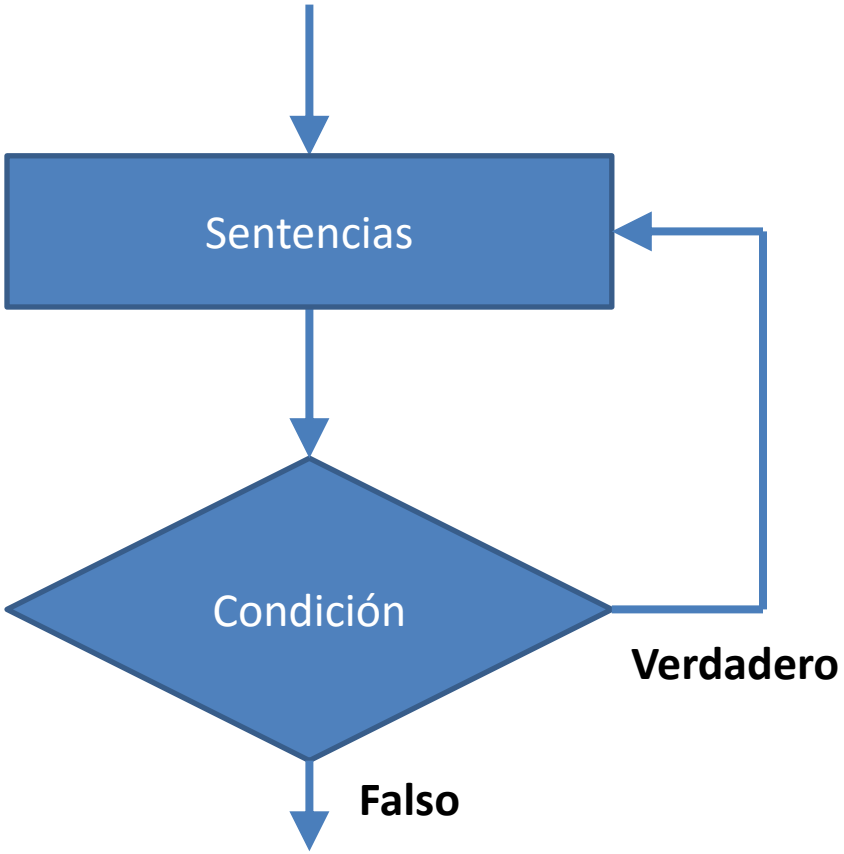
Estructura repetitiva (while)



Mientras (condición) hacer
Sentencias
Fin mientras

Programación

Estructura repetitiva (do while)



hacer
sentencias
mientras (condición)

Programación

Ejemplos

Programación

Diagramas de flujo y pseudocódigo (Ejemplo 1)

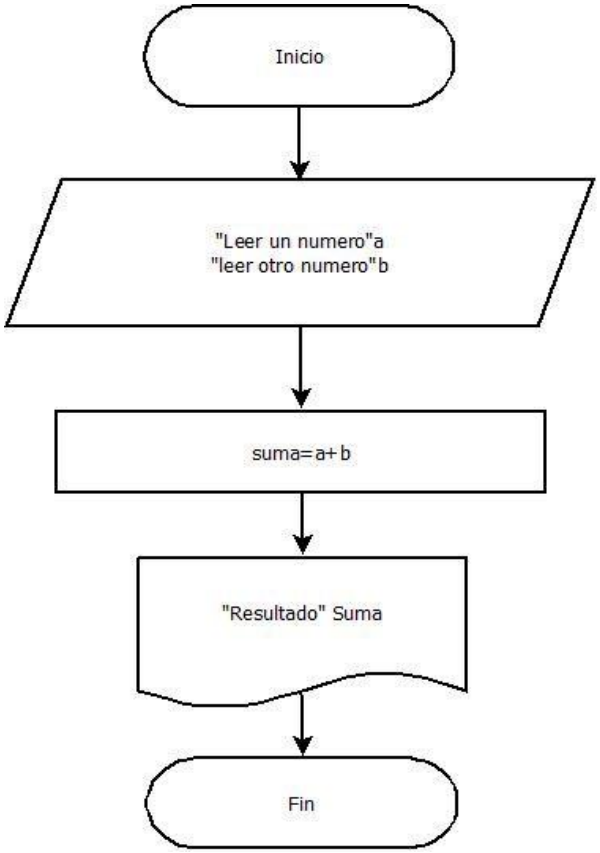


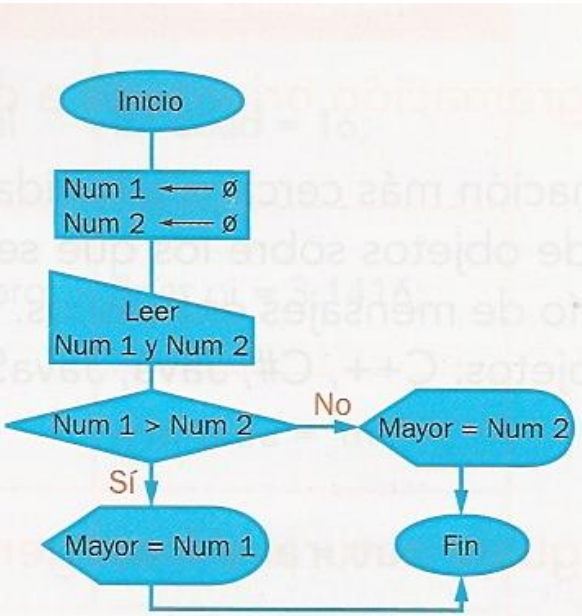
Diagrama de flujo y pseudocódigo correspondiente a un programa que calcula la suma de dos números introducidos por el usuario y muestra el resultado por pantalla.

```
Inicio
    leer (a)
    leer (b)
    suma = a + b
    escribir (suma)
Fin
```

Programación

Diagramas de flujo y pseudocódigo (Ejemplo 2)

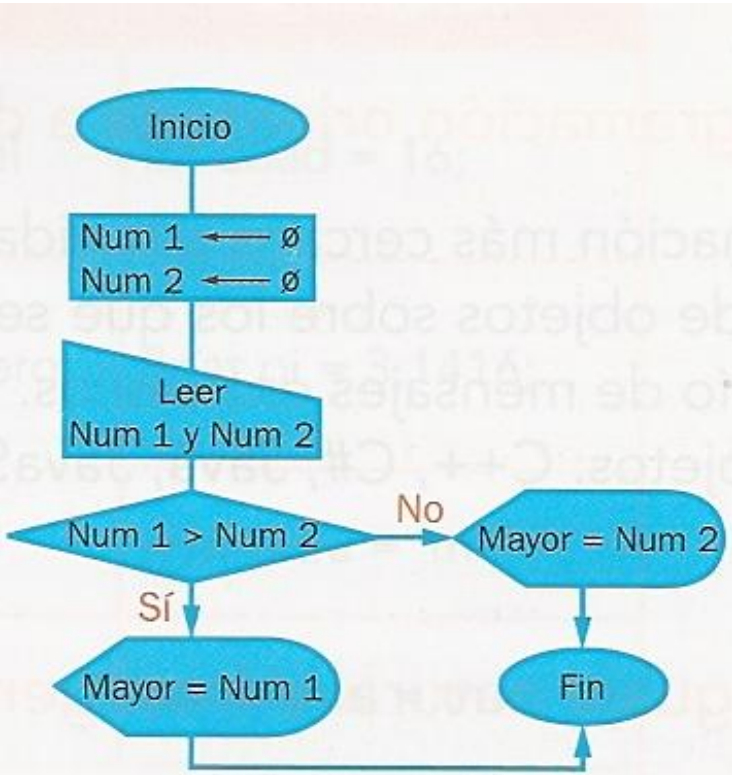
Diagrama de flujo y pseudocódigo de un algoritmo que calcula el mayor de dos números introducidos por el usuario.




```
Inicio
   Num 1 = 0
   Num 2 = 0
   Leer Num 1 y Num 2
   Si Num 1 > Num 2
      "El mayor es" Num1
   Si no
      "El mayor es" Num2
Fin
```

Programación

Diagramas de flujo y pseudocódigo (Ejemplo 2)

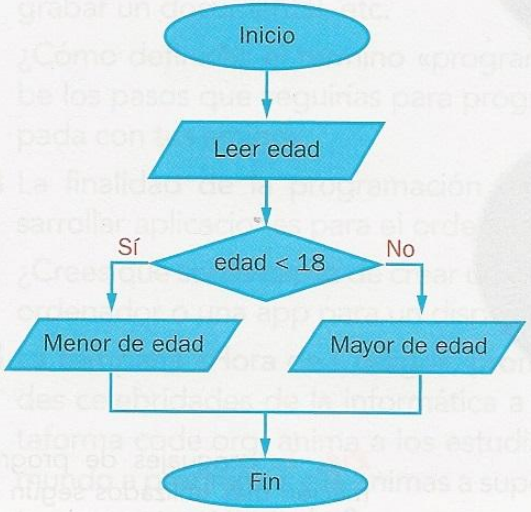



```
Inicio
    Num 1 = 0
    Num 2 = 0
    Leer Num 1 y Num 2
    Si Num 1 > Num 2
        "El mayor es" Num1
    Si no
        "El mayor es" Num2
Fin
```

Programación

Diagramas de flujo y pseudocódigo (Ejemplo 3)

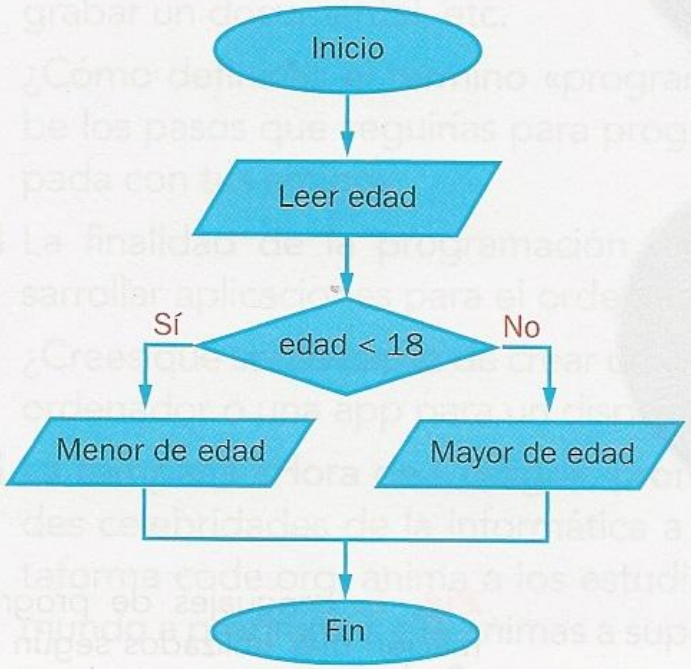
Diagrama de flujo y pseudocódigo de un algoritmo que pide la edad al usuario y muestra por pantalla si es mayor o menor de edad.




```
PROGRAMA edadPersona
Variables
    Entero edad
Inicio
    Escribir "Introduce una edad"
    Leer edad
    Si (edad < 18) entonces
        Escribir "Menor de edad"
    Si no
        Escribir "Mayor de edad"
    Fin si
Fin
```

Programación

Diagramas de flujo y pseudocódigo (Ejemplo 3)




```
PROGRAMA edadPersona
Variables
    Entero edad
Inicio
    Escribir "Introduce una edad"
    Leer edad
    Si (edad < 18) entonces
        Escribir "Menor de edad"
    Si no
        Escribir "Mayor de edad"
    Fin si
Fin
```

Fin