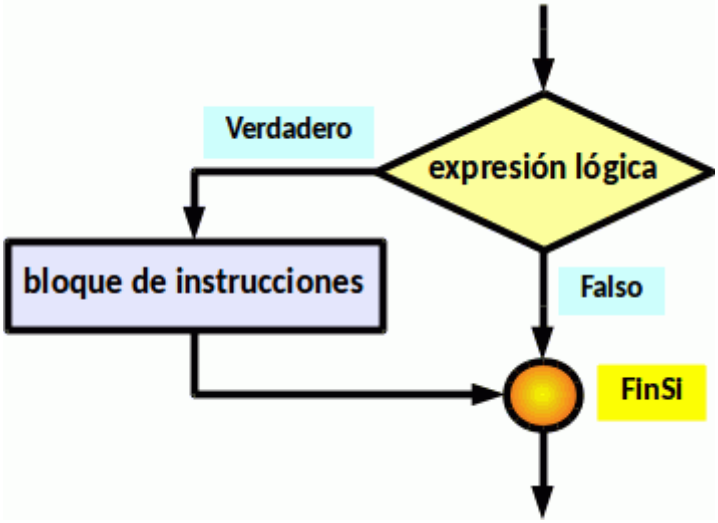
Estructuras de control alternativas

También llamadas de **selección** o **condicionales**. Alteran la secuencia de ejecución según el resultado de evaluar una expresión.

Estructura de control if

Diagrama de flujo



Comportamiento

Se evalúa la expresión lógica contenida entre paréntesis.

- Si es verdadera se ejecutan las sentencias.
- Si es falsa se continua con la siguiente instrucción.

Sintaxis

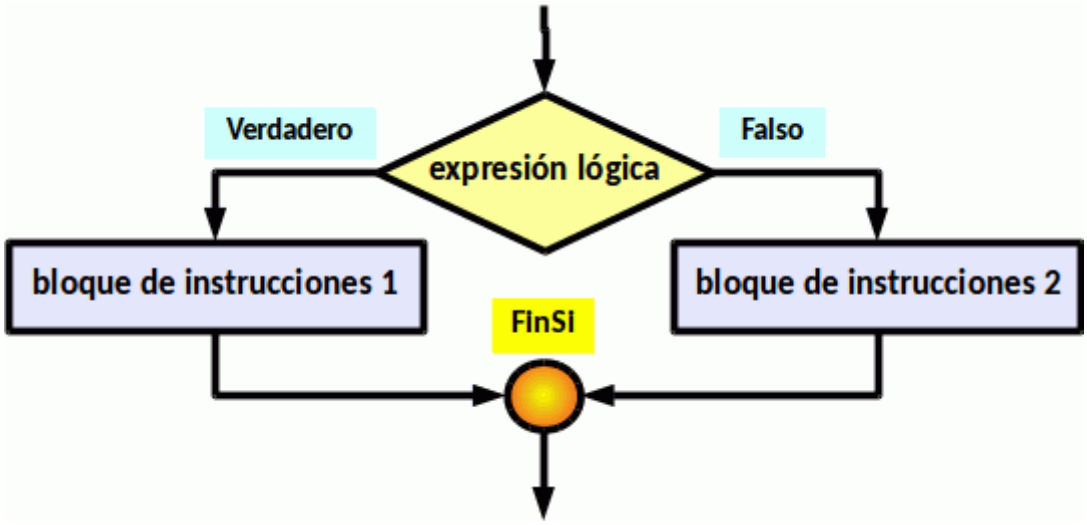
```
if (expresión_lógica) {
    bloque_de_instrucciones;
}
```

Ejemplo

```
if (edad > 18) {
   printf("ADULTO");
   precioEntrada = 20;
}
```

Estructura de control if-else

Diagrama de flujo



Comportamiento

Se evalúa la expresión lógica contenida entre paréntesis.

- Si es verdadera se ejecuta el bloque de código asociado a `if`.
- Si es falsa se ejecuta el bloque de código asociado a `else`.

Sintaxis

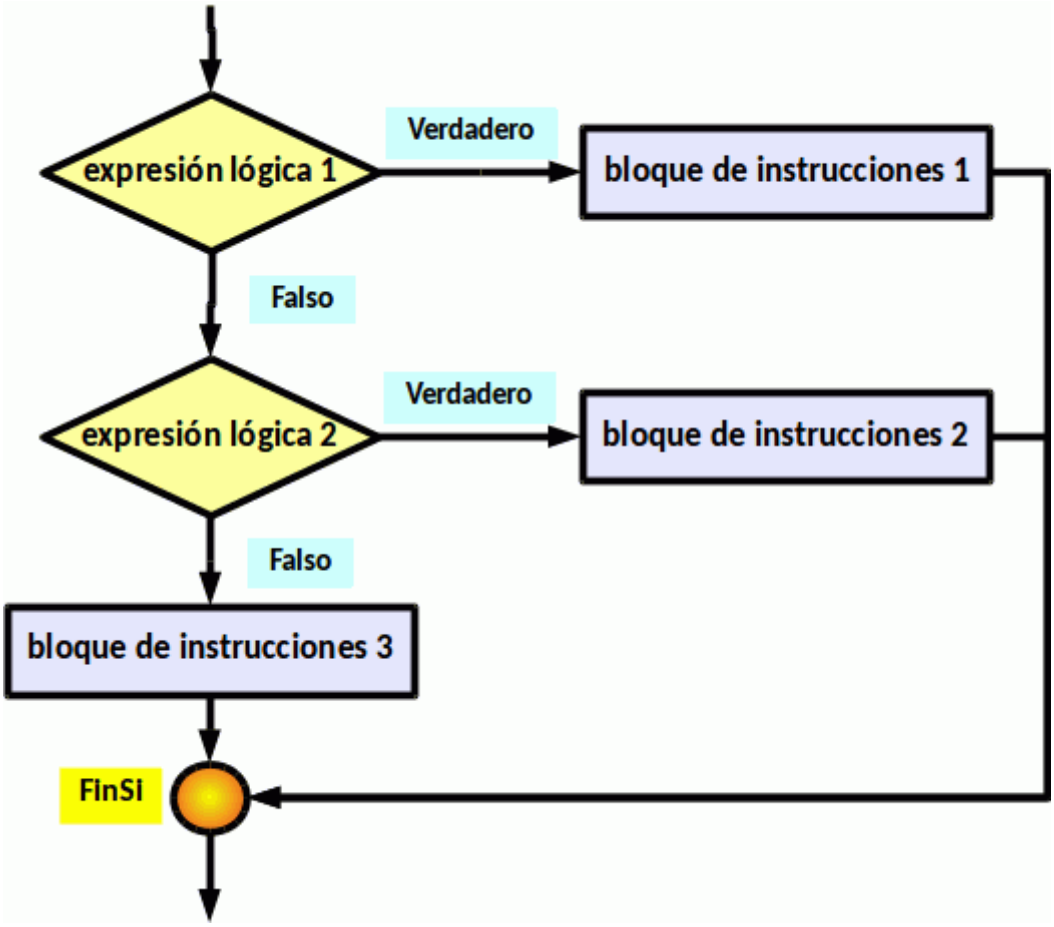
```
if (expresión_lógica) {
    bloque_instrucciones_1;
} else {
    bloque_instrucciones_2;
}
```

Ejemplo

```
if (a > b) {
   printf("A es mayor que B");
} else {
   printf("A es menor o igual a B");
}
```

Estructura de control if anidada

Diagrama de flujo



Comportamiento

- Representan diferentes ejecuciones alternativas y mutuamente excluyentes.
- En caso de que todas las expresiones lógicas sean falsas se ejecutará el último bloque.

Sintaxis

```
if (expresión_lógica_1) {
    bloque_instrucciones_1;
} else if(expresión_lógica_2){
    bloque_instrucciones_2;
} else {
    bloque_instrucciones_3;
}
```

Ejemplo 1 de código. Programa de condiciones anidadas.

ejemplo_nido1.c

```c
#include <stdio.h>

int main(void) {
   int nota; //Variable para almacenar la nota

   printf("Introduzca una nota numerica para el alumno: (0-10) \n");
   scanf("%i", &nota);

   if ((nota >= 0) && (nota < 5)) { // Alternativa 1
      printf("El alumno ha suspendido \n");
   } else if (nota <= 10) { // Alternativa 2
      printf("El alumno ha aprobado \n");
   } else { // Alternativa por defecto
      printf("La nota introducida es incorrecta. \n");
```

```
      printf("Rango valido 0 - 10 \n");
   }
   return 0;
}
```

 *Ejemplo 2 de código*. Programa de condiciones anidadas y no anidadas.

ejemplo_nido2.c

```
/*
En este ejemplo:
 * El Bloque 1 y el Bloque 2 son excluyentes: Si se ejecuta uno el otro no.
 * Los Bloques 3 y 4 son no excluyentes, ejecutándose siempre.
*/
#include <stdio.h>

int main(void) {
   int num1, num2;

   printf("Introduzca el valor de los números: num1 y num2 \n");
   scanf("%d", &num1);
   scanf("%d", &num2);

   if (num1 > num2) {
      printf("Bloque 1: num1 es mayor que num2 \n");
      num2 = num2+10;
      printf("Bloque 1: Ahora num1 vale %d y num2 vale %d \n", num1, num2);
   } else if (num2 > num1) {
      printf("Bloque 2: num2 es mayor que num1 \n");
      num1 = num1+10;
      printf("Bloque 2: Ahora num1 vale %d y num2 vale %d \n", num1, num2);
   }

   if (num1 > num2) {
      printf("Bloque 3: num1 es mayor que num2 \n");
      num2 = num2+10;
      printf("Bloque 3: Ahora num1 vale %d y num2 vale %d \n", num1, num2);
   }

   if (num2 > num1) {
      printf("Bloque 4: num2 es mayor que num1 \n");
      num1 = num1+10;
      printf("Bloque 4: Ahora num1 vale %d y num2 vale %d \n", num1, num2);
   }

   return 0;
}
```

 *Ejercicio propuesto*

Analiza el funcionamiento del programa del ejemplo 2 anterior e indica las condiciones que se tienen que dar para que:

- a) Solo se ejecuten el Bloque1 y el Bloque3.
- b) Solo se ejecuten el Bloque1 y el Bloque4.

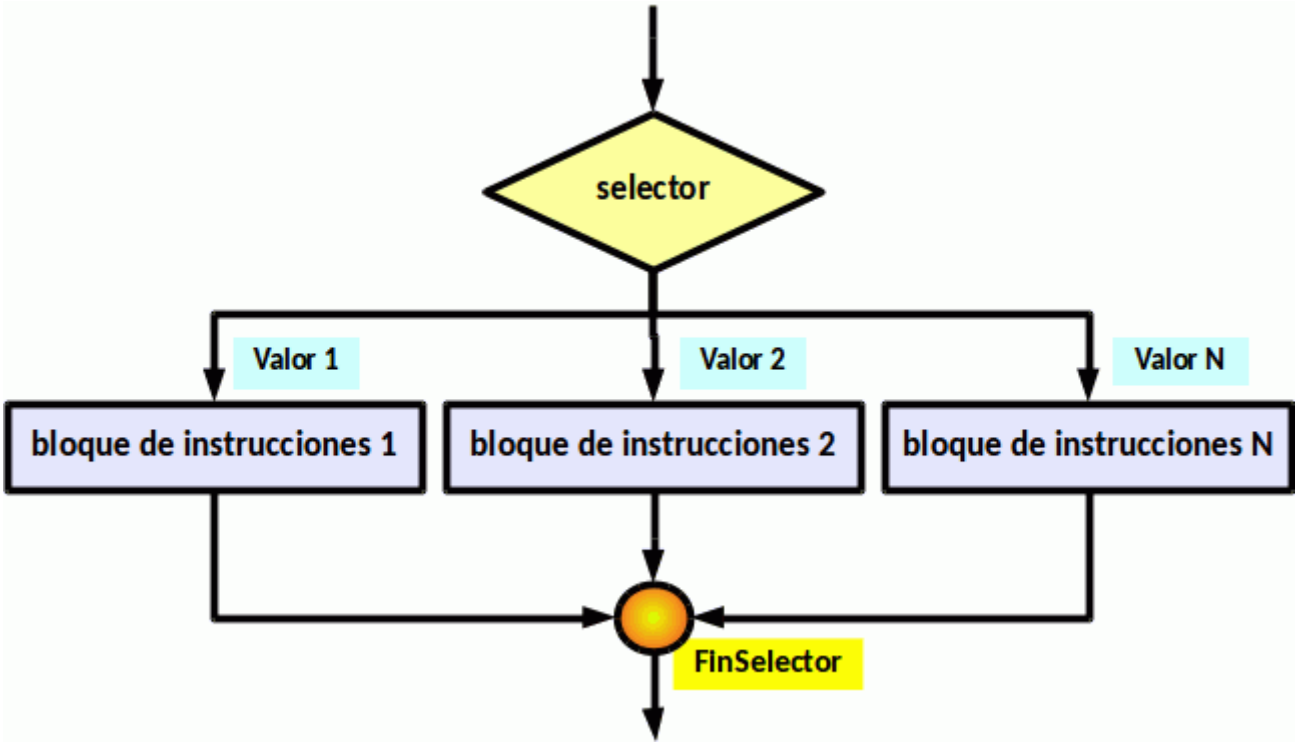
Pon en cada caso un ejemplo de valor para num1 y para num2.


Respuesta

- En el caso (a), num1 debe ser mayor que num2 al menos en 20 unidades. Ejemplo: num1=21 y num2=1.
- En el caso (b), num1 debe ser mayor que num2 como mucho en 9 unidades. Ejemplo: num1=21 y num2=20.

Estructura de control switch

Diagrama de flujo



Comportamiento

Estructura de selección múltiple a partir de una variable selectora.

Sintaxis

```
switch (selector) {
    case valor_1:
        bloque_instrucciones_1;
        break;
    case valor_2:
        bloque_instrucciones_2;
        break;
    ...
    case valor_n:
        bloque_instrucciones_n;
        break;
    default:
        bloque_instrucciones_defecto;
  }
```

El **selector** debe ser una variable o expresión de tipo entera, lógica o carácter. No puede ser una expresión real.

Al finalizar cada bloque se debe incluir la instrucción `break`.

- El efecto de la instrucción es dar por terminada la ejecución de la instrucción `switch`.

- Si se omite la instrucción `break`, se ejecutarán todas las instrucciones del `switch` a partir de ese punto, hasta encontrar una nueva instrucción `break`.

Bloque `default`: Si el valor de la variable selectora no coincide con el valor de algún bloque, se ejecuta el bloque por defecto o `default` si existiese.

 Ejemplo 1 de código Programa que muestra el nombre de un polígono en función del número de lados, utilizando `switch`.

ejemplo_switch1.c

```c
#include <stdio.h>

int main(void) {
   int numeroLados; //Variable para almacenar el valor

   scanf("%i", &numeroLados); //Se lee el número de lados

   switch (numeroLados) { //La variable es el selector
      case 0: case 1: case 2: //Varios posibles valores agrupados
         printf("no es un poligono");
         break;
      case 3:
         printf("triangulo");
         break;
      case 4:
         printf("rectangulo");
         break;
      case 5:
         printf("pentagono");
   }

   return 0;
}
```

 Ejemplo 2 de código Programa para mostrar si una letra introducida por teclado es una vocal, utilizando `switch`.

ejemplo_switch2.c

```c
#include <stdio.h>

int main(void) {
   char c; //Se define la variable

   scanf("%c",&c); // Se lee la variable

   switch (c) {
      case 'A':
         printf("vocal A");
         break;
      case 'E':
         printf("vocal E");
         break;
      case 'I':
         printf("vocal I");
         break;
      case 'O':
         printf("vocal O");
```

```
            break;
        case 'U':
            printf("vocal U");
            break;
        default: //Bloque que se ejecuta si no coincide ningún valor
            printf("consonante");
    }

    return 0;
}
```

Ejercicio propuesto

Analiza el funcionamiento del programa del ejemplo 2 anterior y contesta:

- a) Comprueba si funciona para las vocales minúsculas. ¿Porqué?.
- b) Modifica el programa para que funcione tanto para minúsculas como para mayúsculas.

Respuesta

a) No funciona, pues 'a' y 'A' son diferentes caracteres.

b) Una posible solución sería:

ejemplo_switch2b.c

```
#include <stdio.h>

int main(void) {
    char c; //Se define la variable

    scanf("%c",&c); // Se lee la variable

    switch (c) {
        case 'a': case 'A':
            printf ("vocal A");
            break;
        case 'e': case 'E':
            printf ("vocal E");
            break;
        case 'i': case 'I':
            printf ("vocal I");
            break;
        case 'o': case 'O':
            printf ("vocal O");
            break;
        case 'u': case 'U':
            printf ("vocal U");
            break;
        default: //Bloque que se ejecuta si no coincide ningún valor
            printf ("consonante");
    }

    return 0;
}
```