Estructuras selectivas

Estructuras de control

Intervienen en la ejecución secuencial.

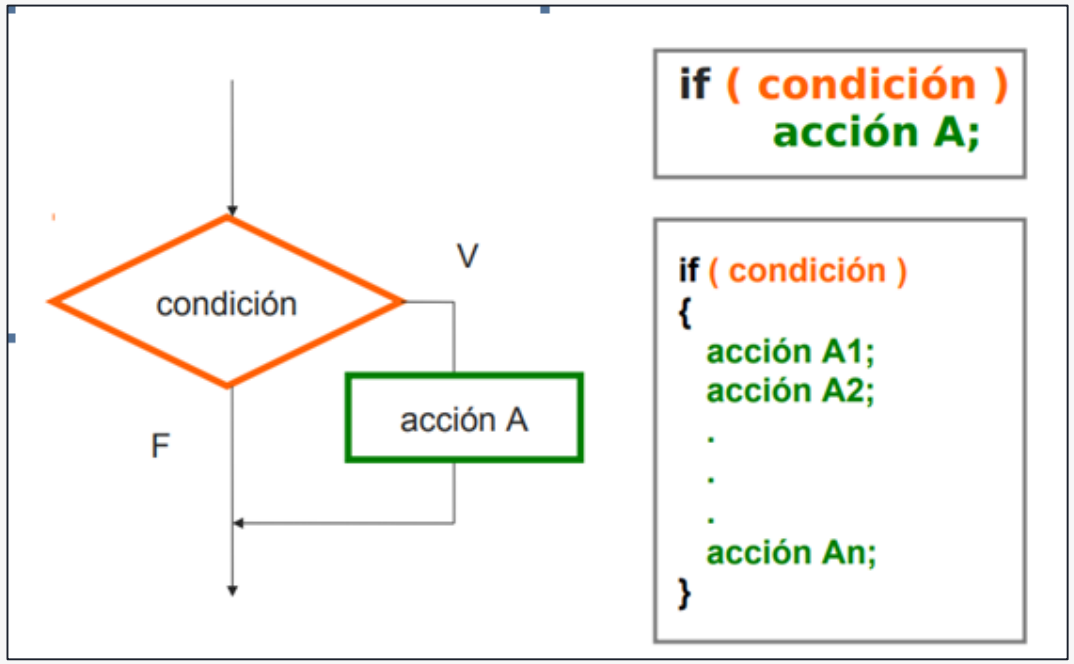
Permiten:

- Controlar la forma en que se ejecutan las sentencias del programa,
- Evaluar condiciones para decidir si ejecutar o no una sección del programa.

A continuación, distintos tipos de condicionales.

Condicional "`if`"

Estructuras de control simple




```
    if (condicion)
    {
        // Bloque de código que se ejecuta si la condición se cumple
    }

// Siguiente línea a ejecutar
```

Ejemplo "`if`"

Estructuras de control simple

Supongamos que obtenemos un entero mediante la función scanf y queremos verificar si el mismo es mayor a cero:

```
    if (numero>0)
    {
        printf("El numero ingresado es mayor a cero\n");
    }

// Siguiente línea a ejecutar
```

En este ejemplo, cuando se ingresa un número menor o igual a cero, no se imprime nada.

Condicional "`if-else`"

Estructuras de control simple

Sintaxis simple:

```
if (condición)
    // Sentencia que se ejecutará si la condición es VERDADERA
else
    // Sentencia que se ejecutará si la condición es FALSA
```

- El else es optativo
- Cada else se "acopla" al if más cercano
- Si aplico más de una sentencia por verdadero o falso deben usarse llaves para agrupar las sentencias.

Condicional "`if-else`"

Estructuras de control simple

Sintaxis con llaves:

```
if (condición) {
    sentencia1 por verdadero;
    sentencia2 por verdadero;
}
else {
    sentencia1 por falso;
    sentencia2 por falso;
}
```

- Es una buena práctica **usar siempre** esta sintaxis con llaves, aún cuando haya una sola sentencia.

Ejemplo "if-else"

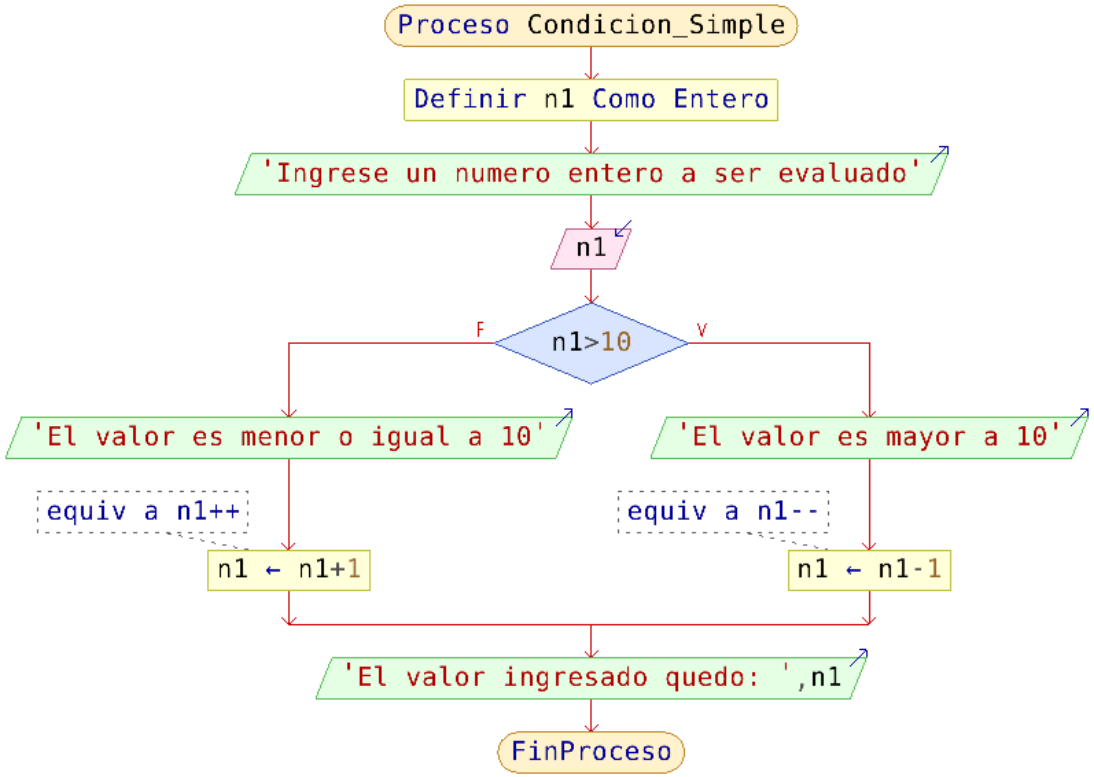
Estructuras de control simple

El programa deber evaluar numero ingresado.

Si el numero ingresado es mayor a 10 notificarlo y restarle 1:

Caso contrario notificarlo y sumarle 1

Al final imprimir como queda el numero ingresado.



Solución en C

```c
#include<stdio.h>

/* El programa deber evaluar numero ingresado. */
/* Si el numero ingresado es mayor a 10 notificarlo y restarle 1:  */
/* Caso contrario notificarlo y sumarle 1 */
/* Al final imprimir como queda el numero ingresado. */

int main() {
    int n1;
    printf("Ingrese un numero entero a ser evaluado\n");
    scanf("%d",&n1);
    if (n1>10) {
        printf("El valor es mayor a 10\n");
        n1 = n1-1; /* equiv a n1-- */
    } else {
        printf("El valor es menor o igual a 10\n");
        n1 = n1+1; /* equiv a n1++ */
    }
    printf("El valor ingresado quedo: %i\n",n1);
    return 0;
}
```

Condicional "`if-else if`"

Estructuras de control múltiple

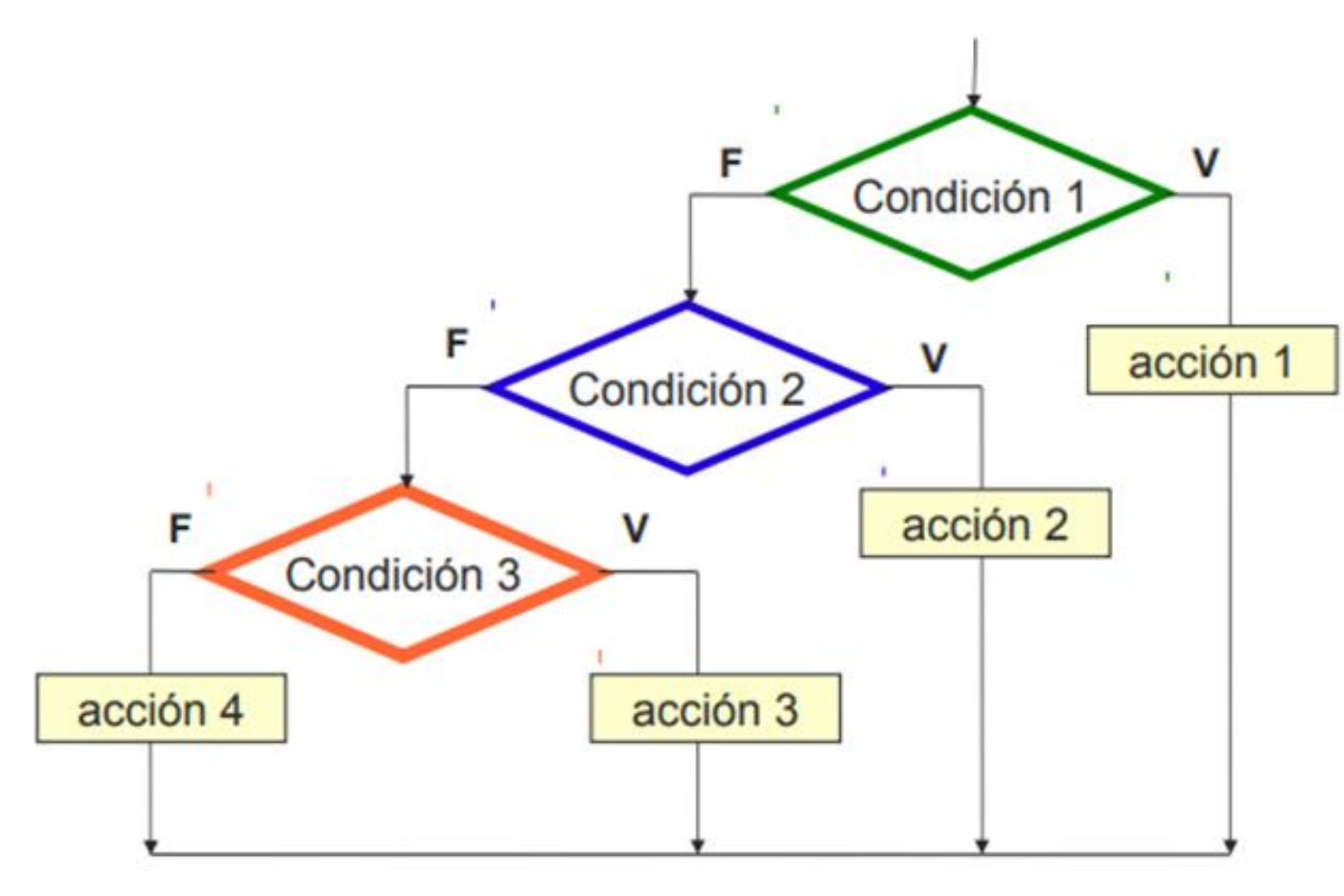
Sintaxis simple:

- Más de una estructura "if" relacionadas
- Si no se cumple la primera condición, se comprueba la segunda y así sucesivamente
- En el siguiente ejemplo **sólo** se ejecutará un bloque de sentencias

```
if (condición_A)
    { Bloque de sentencias que se ejecuta si se cumple la condición A}
else if (condición_B)
    { Bloque de sentencias que se ejecuta si se cumple la condición B }
else if (condición_C)
    { Bloque de sentencias que se ejecuta si se cumple la condición C }
                                        else
        { Sentencias que se ejecutan si no se cumplen las condiciones
anteriores }
```

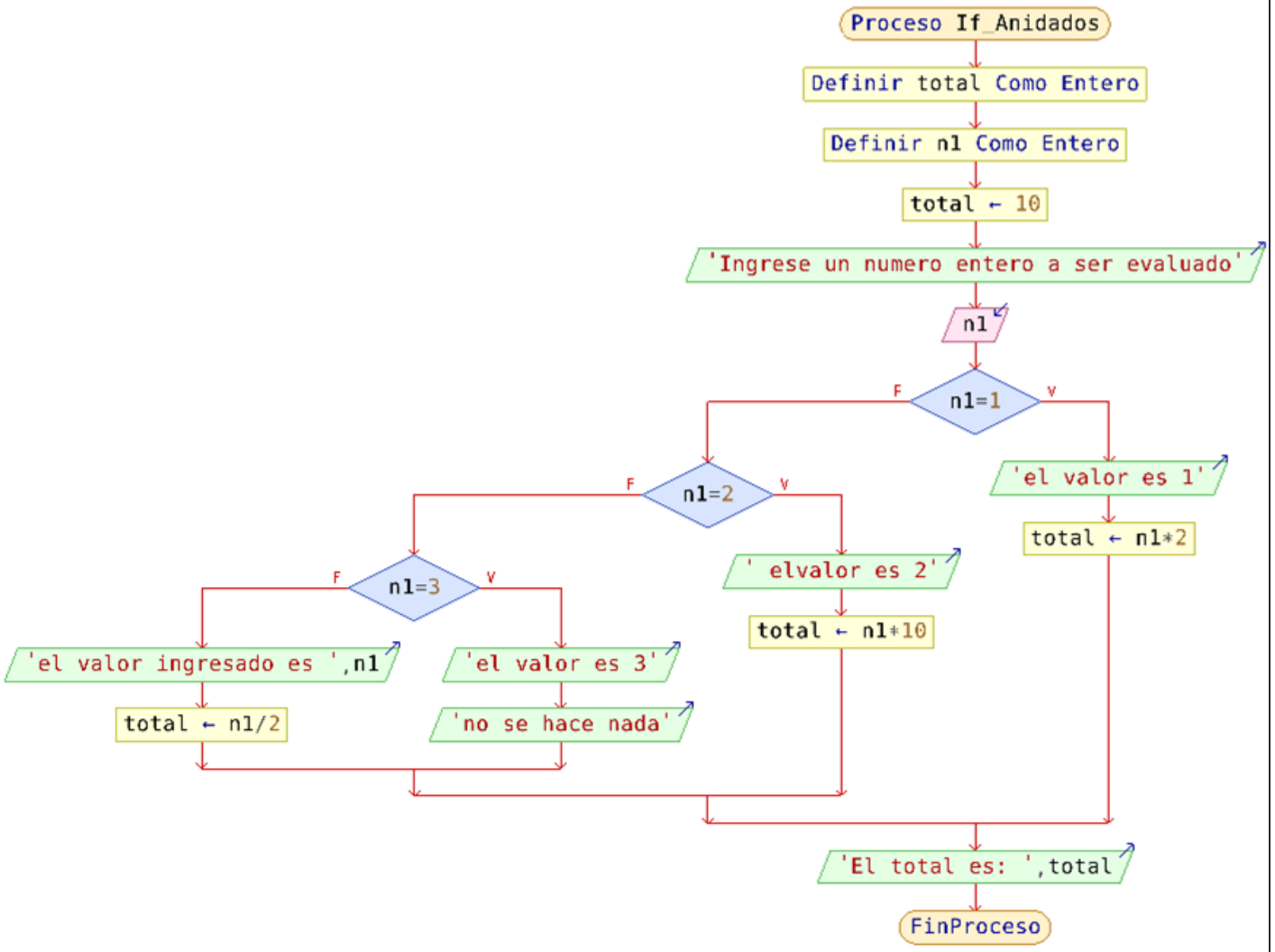
Diagrama de ejemplo "`if-else if`"

Estructuras de control múltiple



Ejemplo "`if-else if`"

Estructuras de control múltiple



Solución en C

```c
int main()
{
    int n1;
    int total;
    total = 10;
    printf("Ingrese un numero entero a ser evaluado\n");
    scanf("%i",&n1);
    if (n1==1)
    {
        printf("el valor es 1\n");
        total = n1*2;
    }
    else if (n1==2)
    {
        printf(" elvalor es 2\n");
        total = n1*10;
    }
    else if (n1==3)
    {
        printf("el valor es 3\n");
        printf("no se hace nada\n");
    }
    else
    {
        printf("el valor ingresado es %i\n",n1);
        total = n1/2;
    }
    printf("El total es: %i\n",total);
    return 0;
}
```

Más de una condición

Estructuras de control múltiple

Una o más condiciones con *AND* y *OR* :

```
if (condición_1 y condición_2)
    { bloque de sentencias A }
else if (condición_2 ó condición_3)
    { bloque de sentencias B }
else
    { bloque de sentencias restantes}
```

- **and** = Ambas condiciones verdaderas
  - Operador: **&&**
- **or** = Al menos una condición verdadera
  - Operador: **| |**

Los operadores “and” y ”or” pueden utilizarse en las estructuras “if”, “if else” y en “if else if”

Más de una condición

Estructuras de control múltiple

p	q	p && q
F	F	F
F	V	F
V	F	F
V	V	V

AND (y)

p	q	p || q
F	F	F
F	V	V
V	F	V
V	V	V

OR (ó)

Ejemplo

Estructuras de control múltiple

```c
#include <stdio.h>

int main()
{
    int numero;
    printf ("Ingrese un numero: ");
    scanf("%d", &numero);

    if (numero>0 && numero<100)
    {
        printf("El valor esta entre cero y 100");
    }
    else if (numero<=0 || numero>=100)
    {
        printf("El valor es menor o igual a cero o mayor o igual a 100");
    }
    return 0;
}
```

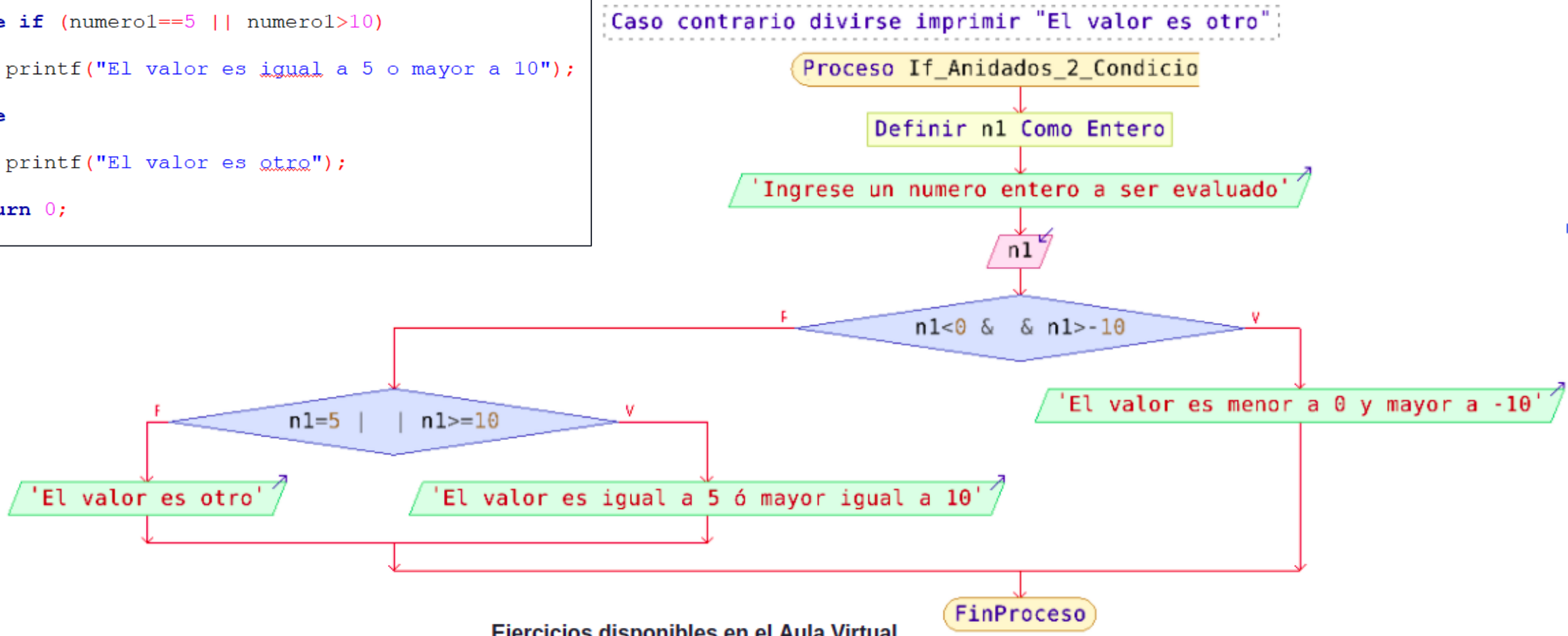
Solución en C

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    int numero1;
    printf ("Ingrese el valor numerico: ");
    scanf("%d", &numero1);

    if (numero1<0 && numero1>-10)
    {
        printf("El valor es menor a 0 y mayor a -10");
    }
    else if (numero1==5 || numero1>10)
    {
        printf("El valor es igual a 5 o mayor a 10");
    }
    else
    {
        printf("El valor es otro");
    }
    return 0;
}
```

El programa debe solicitar un valor y evaluar:
Si el valor ingresado es menor a 0 y mayor a -10. Imprimir mensaje apropiado.
Si el valor ingresado es igual a 5 o mayor igual a 10. Imprimir mensaje apropiado.
Caso contrario imprimir “El valor es otro”



Ejercicios disponibles en el Aula Virtual

Operador ternario

Estructuras de control

Condicional en una sola sentencia :

- Similar al `if-else`
- Asigna un valor u otro dependiendo de una condición

```
mayor = numA > numB ? numA : numB;
```

Equivale a :

```
if (numA > numB)
{
    mayor = numA;
}
else
{
    mayor = numB;
}
```

Ejemplo en C

```
int main()
{
    int numA, numB, mayor;
    printf("Ingrese valor a para numA: \n");
    scanf("%d", &numA);
    printf("Ingrese valor a para numB: \n");
    scanf("%d", &numB);
    mayor = numA>numB ? numA : numB;
    printf("El valor mayor es: %d \n", mayor);
    return 0;
}
```

Ejemplo de operador ternario

Estructuras de control

Facultad de INGENIERÍA UNLZ

También ejecuta:

```c
#include <stdio.h>

int main() {
    int num;

    printf("Ingrese un número entero: ");
    scanf("%d", &num);

    num >= 0 ? printf("El número es positivo.\n") : printf("El número es negativo.\n");

    return 0;
}
```

Por lo tanto, es especialmente útil para:

- Realizar asignaciones
- Imprimir un mensaje

Ejemplo de operador ternario

Estructuras de control

Caso de uso: Determinar si alumno está aprobado

- Con operador ternario:

```
…
esta_alumno_aprobado = nota>=4 ? 1 : 0;
…
```

- Sin operador ternario:

```
…
if (nota >= 4)
    esta_alumno_aprobado = 1;
else
    esta_alumno_aprobado = 0;
…
```

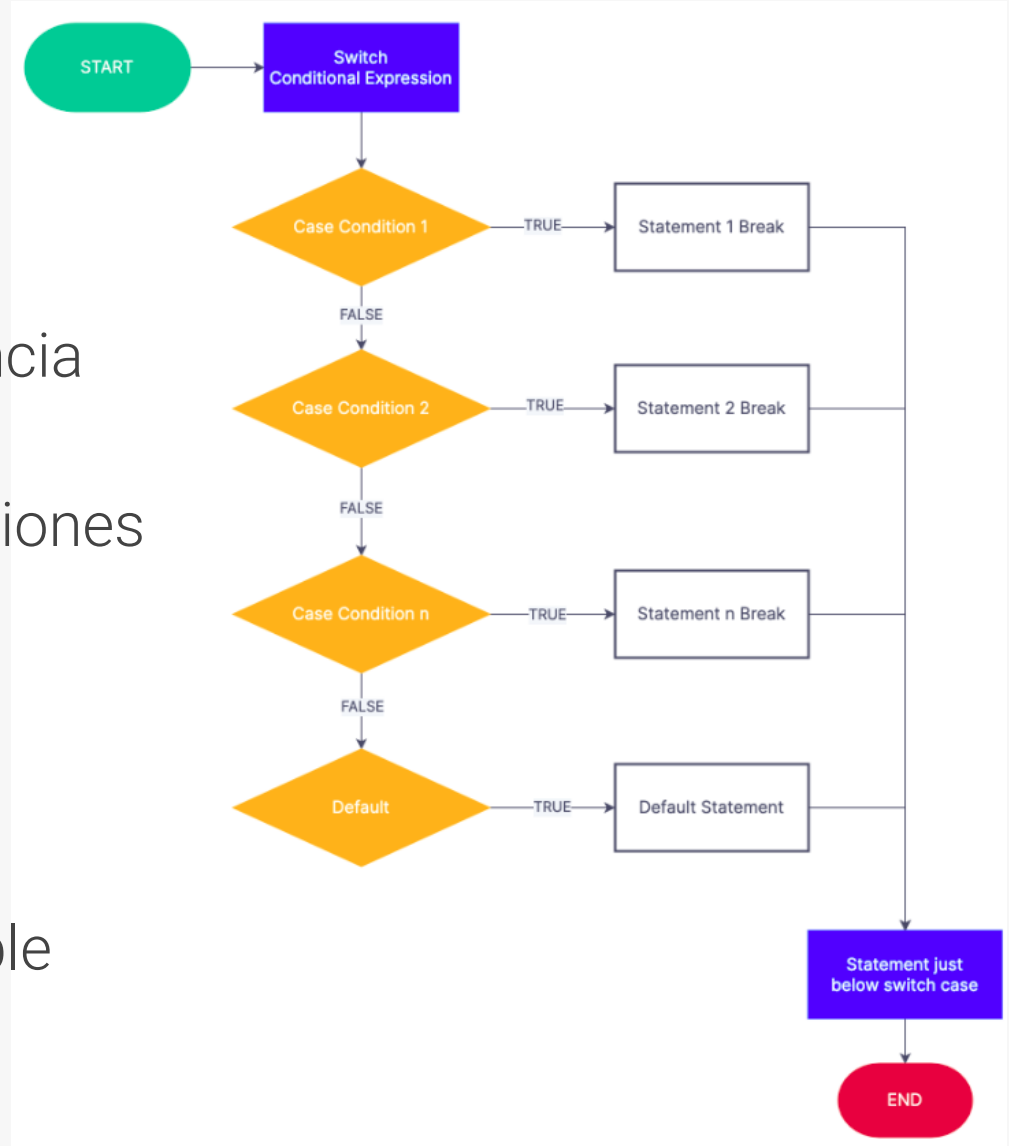
Ambas formas son válidas y las elige el programador por comodidad

Selector "`switch case`"

Estructuras de control múltiple

Un valor, muchos caminos:

- Depende de una variable o sentencia
- Permite elegir entre múltiples opciones
  - También llamadas *casos*
- Permite opción "por defecto"
  - Llamada *default*
  - Cuando ninguna otra se cumple



Sintaxis de "`switch case`"

Estructuras de control múltiple

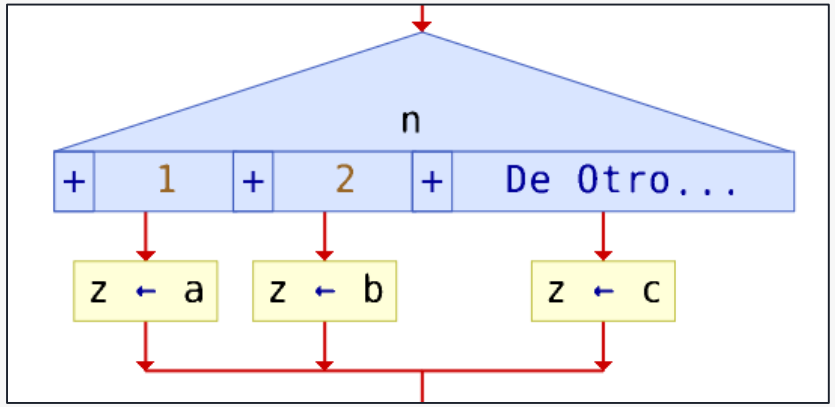
- Evalúa una vez
- Compara cada caso ó default
- Termina en **`break;`**

```
switch (expresión) {
case valor_1:
        sentencias1
       break;
case valor_2:
        sentencias2
        /* Sin break continúa la ejecución con el código
           a continuación de la siguiente etiqueta*/
default:
        sentencias por defecto (caso contrario)
}
```

- Sin **`break;` continúa** (posible comportamiento no deseado)
- Código limpio y estructurado.

Más sobre "`switch case`"

Estructuras de control múltiple



Se puede usar el operador `...` para indicar un rango numérico

```
switch (num) {
    case 0:
        printf("El número es cero.\n");
        Break;
    case 1 ... 5:
        printf("El número esta entre 1 y 5.\n");
        Break;
    default:
        printf("El número no es 0, ni esta entre 1 y 5\n");
        Break;
}
```

Ejemplo en C:

Dependiendo el valor de **n** asignar a **z** el valor de **a**, **b** o **c**.

```
switch (n) {
    case 1:
        z = a;
        break;
    case 2:
        z = b;
        break;
    default:
        z = c;
}
```

Recordar utilizar "`break`" al final de cada "`case`"

Ejemplo "`switch case`"

Estructuras de control múltiple

```c
#include <stdio.h>
int main()
{
    /* Escribe el día de la semana */
    int dia;
    printf("Introduce el numero de dia de la semana: ");
    scanf("%d",&dia);
    switch(dia){
      case 1:
          printf("Lunes");
          break;
      case 2:
          printf("Martes");
          break;
      case 3:
          printf("Miercoles");
          break;
      case 4:
          printf("Jueves");
          break;
      case 5:
          printf("Viernes");
          break;
      case 6:
          printf("Sabado");
          break;
      case 7:
          printf("Domingo");
          break;
      default:
        printf("Error, numero invalido");
    }
    return 0;
}
```

Enumeradores

Estructuras

Mejoran la legibilidad y optimizan el código al:

- Definir **constantes** y un nuevo **tipo** de dato de una vez.
- Número enteros, desde cero y se incrementan

```
enum dia {LUNES, MARTES, MIERCOLES, JUEVES, VIERNES, SABADO, DOMINGO};
```

- Permiten declarar variable del tipo "`enum dia`" y ahorrar memoria

```
enum dia hoy, ayer;
hoy  = LUNES;  //asigna 0 a hoy
ayer = JUEVES; //asigna 3 a ayer
```

- Se puede interferir en el orden predefinido

```
enum operaciones {LEER = 1, ESCRIBIR, VERIFICAR, BORRAR = 8, ALERTAR};
```

Ejemplo

Enumeradores con switch

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
    enum opciones {INGRESAR=0, LISTAR, ELIMINAR, SALIR};
    int valor_ingresado = 0;

    printf("Ingrese Opcion: ");
    scanf("%d", &valor_ingresado);

    switch (valor_ingresado)
    {
    case INGRESAR:
        printf("Ingresando... \n");
        break;
    case LISTAR:
        printf("Listando... \n");
        break;
    case ELIMINAR:
        printf("Eliminando... \n");
        break;
    case SALIR:
        printf("Saliendo... \n");
        break;
    default:
        printf("Opcion incorrecta \n");
        break;
    }

    return 0;
}
```