Estructuras de Control de Selección en C

Estructuras de Control de Selección en C

Las estructuras de control de selección permiten que un programa tome decisiones y ejecute diferentes bloques de código según condiciones específicas.

En C, las principales estructuras de selección son:

1. **Condicionales if, else, else if**
2. **La estructura switch**

Condicionales if, else y else if

Las condicionales *if, else y else if* permiten ejecutar instrucciones dependiendo del resultado de una condición. Se usan cuando es necesario evaluar expresiones booleanas (verdadero o falso).

Estructura básica de if

Estructura básica

```
if (condición)
{
// Código a ejecutar si
//la condición es verdadera
 }
```

Ejemplo

```
int edad = 18;
 if (edad >= 18)
{
 printf("Eres mayor de edad.\n");
 }
```

Estructura básica de if-else

Estructura

```
if (condición) {
// Código si la condición es verdadera
}
else {
// Código si la condición es falsa
}
```

Ejemplo

```
int numero = 10;
if (numero % 2 == 0) {
printf("El número es par.\n");
}
else {
printf("El número es impar.\n");
}
```

Estructura de else if para múltiples condiciones

Estructura

```
if (condición1) {
// Código si la condición1 es
verdadera
} else if (condición2) {
 // Código si la condición2 es
verdadera
} else {
 // Código si ninguna de las
condiciones anteriores es verdadera
}
```

Ejemplo

```
int calificacion = 85;
if (calificacion >= 90)
{
printf("Tu calificación es A.\n");
} else if (calificacion >= 80)
{
printf("Tu calificación es B.\n");
} else if (calificacion >= 70)
{
printf("Tu calificación es C.\n");
} else {
printf("Reprobaste.\n");
}
```

Operadores lógicos y relacionales en if

Operador	Descripción	Ejemplo (a = 5, b = 10)
==	Igualdad	a == b → false
!=	Diferente	a != b → true
<	Menor que	a < b → true
>	Mayor que	a > b → false
<=	Menor o igual	a <= b → true
>=	Mayor o igual	a >= b → false

Operadores lógicos (combinar múltiples condiciones)

Operador	Descripción	Ejemplo (a = 5, b = 10, c = 15)
&&	AND (y) - ambas deben ser verdaderas	(a < b) && (b < c) → true
||	OR (o) - alguna de las dos debe ser validada	(a < b) || (b < c) → true
!	NOT (negación) - invierte el valor booleano	!(a > b) → true

Ejemplo

```
int edad = 20;
char tieneLicencia = 'S';
if (edad >= 18 && tieneLicencia == 'S') {
    printf("Puedes conducir.\n");
} else {
    printf("No puedes conducir.\n");
}
```

Estructura switch

La estructura switch permite evaluar una variable y ejecutar diferentes bloques de código según su valor. Se usa cuando se tienen múltiples opciones predefinidas en lugar de múltiples if-else.

Estructura de uso de switch

```
switch (expresión) {
    case valor1:
        // Código a ejecutar si expresión == valor1
        break;
    case valor2:
        // Código a ejecutar si expresión == valor2
        break;
    default:
        // Código a ejecutar si no coincide con ningún caso
}
```

Ejemplo

```
int opcion = 2;
switch (opcion) {
    case 1:
        printf("Seleccionaste opción 1.\n");
        break;
    case 2:
        printf("Seleccionaste opción 2.\n");
        break;
    case 3:
        printf("Seleccionaste opción 3.\n");
        break;
    default:
        printf("Opción no válida.\n");
}
```

¿Cuándo usar if y cuándo usar switch?

Criterio	if-else	switch
Condiciones complejas (comparaciones, rangos, expresiones lógicas)	✔️ Sí	❌ No
Comparaciones con valores específicos (constantes, caracteres, enteros)	✔️ Sí	✔️ Sí
Evaluación de una sola variable contra múltiples valores predefinidos	❌ No	✔️ Sí

Recomendaciones

Usa if-else cuando las condiciones involucren rangos de valores o expresiones lógicas.

Usa switch cuando solo necesites comparar una variable con múltiples valores constantes y mejorar la legibilidad del código.

Referencias

Deitel, P. J., & Deitel, H. (2016). C: How to Program (8th ed.). Pearson.

Kernighan, B. W., & Ritchie, D. M. (1988). The C Programming Language (2nd ed.). Prentice Hall.

King, K. N. (2008). C Programming: A Modern Approach (2nd ed.). W. W. Norton & Company.

Stroustrup, B. (2013). Programming: Principles and Practice Using C++ (2nd ed.). Addison-Wesley. (Aunque enfocado en C++, contiene fundamentos de C).