



Programación en C++

MSc. Víctor Melchor Espinoza
cpp.ctic@gmail.com

Unidad

2

Sesión 3: Estructuras Condicionales Múltiples

Contenidos

- a) if múltiple y anidado
- b) switch case

Aprendizajes esperados

- Aprende a evaluar más de una condición.

7.5 Menús

- **Programa controlado por Menú:** ejecución del programa controlado por el usuario que selecciona a partir de una lista de opciones.
- **Menu:** lista de opciones en pantalla.
- Puede ser implementado usando declaraciones **`if/else if`**.

Organización de un programa controlado por Menú

- Muestra la lista de opciones numéricas o literales para ejecutar acciones.
- Se ingresa la selección del usuario.
- Probar la selección del usuario en (*expresion*)
 - Si hay una correspondencia, entonces se ejecuta el código para llevar a cabo la acción deseada.
 - Si no, entonces probar con la siguiente (*expresion*)

7.6 Sentencias `if` Anidadas

- Una sentencia `if` que es parte de la cláusula `if` o `else` de otra sentencia `if`
- Puede ser usada para evaluar más de un dato o condición

```
if (puntaje < 100)
{
    if (puntaje > 90)
        nota = 'A';
}
```

Notas acerca de `ifs` anidados

- Un `else` corresponde al `if` más cercano que no tiene un `else`

```
if (puntaje < 100)
    if (puntaje > 90)
        nota = 'A';
    else ...    // va con el segundo if
                // no con el primero
```

- La indentación adecuada ayuda a la comprensión

7.7 Operadores Lógicos

- Usados para crear expresiones relacionales a partir de otras expresiones relacionales.

Operadores, Significado, y Explicación

&&	AND	La nueva expresión relacional es true si ambas expresiones son true.
 	OR	La nueva expresión relacional es true if al menos una expresión es true.
!	NOT	Invierte el valor de una expresión; expresión true se convierte en false, and false se convierte en true.

Ejemplos de Operador Lógico

```
int x = 12, y = 5, z = -4;
```

<code>(x > y) && (y > z)</code>	<code>true</code>
<code>(x > y) && (z > y)</code>	<code>false</code>
<code>(x <= z) (y == z)</code>	<code>false</code>
<code>(x <= z) (y != z)</code>	<code>true</code>
<code>! (x >= z)</code>	<code>false</code>

Precedencia Lógica

Superior	!
↓	&&
Inferior	

Ejemplo:

(2 < 3) || (5 > 6) && (7 > 8)



es true pues AND se realiza antes de OR

Más sobre Precedencia

Superior	Operadores aritméticos
↓	Operadores relacionales
Inferior	Operadores lógicos

Ejemplo:

$8 < 2 + 7 \parallel 5 == 6$ es true

7.8 Verificar Rangos Numericos con Operadores Lógicos

- Usado para evaluar si un valor está dentro de un rango

```
if (nota >= 0 && nota <= 20)
    cout << "Nota válida";
```

- Puede probar también si un valor cae fuera de un rango

```
if (nota <= 0 || nota >= 20)
    cout << "Nota incorrecta";
```

- No puede usar notación matemática

```
if (0 <= nota <= 20) //Incorrecto
```

7.9 Validando la entrada del Usuario

- **Validación de la entrada:** inspeccionar los datos de entrada para determinar si ésta es aceptable.
- Busca evitar aceptar entradas incorrectas
- Puede realizar varias pruebas:
 - Rango
 - Validar opción del menú.
 - Dividir por cero.

7.10 Más Acerca de Definiciones de Variable y Ambito

- El **Ambito** de una variable es el bloque en el cual ésta está definida, a partir del punto de definición hacia el final del bloque.
- Usualmente se definen al iniciar una función.

Mas Acerca de Definiciones de Variable y Ambito

- Las Variables definidas dentro de { } tienen **ámbito local** o **de bloque**
- Cuando se tiene un bloque anidado dentro de otro bloque, se pueden definir variables con el mismo nombre que en el bloque exterior.
 - Cuando se está en el bloque interior, la definición exterior no está disponible
 - Su uso no es una buena idea.

7.11 Comparando Caracteres y Strings

- Puede usar operadores relacionales con caracteres y objetos string,
`if (Nombre < "Bety")`
- Comparar caracteres es realmente comparar los valores ASCII de los caracteres.
- Comparar los objetos string es comparar los valores ASCII en las cadenas. La comparación se hace caracter por caracter.

7.12 El Operador Condicional

- Puede usarlo para crear **proposiciones** cortas **if/else**.
- Formato: **expr ? expr : expr ;**

First Expression:
Expression to
be tested.



`x < 0`

`?`

`y = 10`

`:`

`z = 20 ;`

3rd Expression:
Execute if the 1st
expression is false.



2nd Expression:
Executes if the 1st
expression is true.



7.13 La Sentencia **switch**

- Usada para seleccionar entre sentencias a partir de varias alternativas.
- En algunas ocasiones puede ser usada en vez de sentencias **if/else if**

Formato proposición **switch**

```
switch (expresion)  
{  
    case exp1: bloque1;  
    case exp2: bloque2;  
    ...  
    case expn: bloquen;  
    default:   bloquen+1;  
}
```

Requisitos de la proposición **Switch**

- 1) ***expresion*** debe ser una variable **char** o una variable entera o una expresión que se evalúa a un valor entero.
- 2) desde ***exp1*** hasta ***expn*** deben ser expresiones enteras constantes y deben ser únicas en la proposición **switch**
- 3) **default** es opcional pero es recomendada.

Como trabaja la proposición **switch**

- 1) Se evalúa *expresion*
- 2) El valor de *expresion* es comparada desde *exp1* hasta *expn*.
- 3) If *expresion* corresponde con el valor *exp*i**, el programa ejecuta la(s) sentencia(s) que siguen a *exp*i** y continúa al final del **switch**
- 4) Si no corresponde con ningún valor, el programa ejecuta las sentencias que setán después de **default**:

La cláusula **break**

- Usada para detener la ejecución en el bloque actual.
- También se usa para salir de una proposición **switch**.
- Util para ejecutar una sola sentencia **case** sin ejecutar las sentencias que lo siguen.

Ejemplo de Sentencia **switch**

```
switch (genero)
{
    case 'f': cout << "femenino";
               break;
    case 'm': cout << "masculino";
               break;
    default : cout<< "género incorrecto";
}
```

Usando **switch** con un Menú

- La proposición **switch** es una opción natural para un programa manejado por menú.
 - Mostrar el menú.
 - Obtener la entrada del usuario.
 - Usar la entrada del usuario como **expresion** en la sentencia **switch**.
 - Usar las opciones del menú como **exp** para ir evaluando cada cláusula **case**.