

Estructuras de Control Repetitivo

Rodolfo Christian Catunta Uturunco (Elemental Bolivia)



24 de febrero de 2025

Contenido

- 1 Sentencia de Iteración: while
- 2 Sentencia de Iteración: do-while
- 3 Sentencia de Iteración: for
- 4 Sentencias Break y Continue

Contenido

- 1 Sentencia de Iteración: while
- 2 Sentencia de Iteración: do-while
- 3 Sentencia de Iteración: for
- 4 Sentencias Break y Continue

Sentencia de Iteración while

La sentencia de iteración while permite ejecutar un set instrucciones, **mientras** se cumpla una condición, es decir el set de instrucciones se ejecuta de forma repetitiva.

```
1 // La condicion de repeticion es una expresion booleana
2 while(condicion_de_repeticion){
3     // Set de instrucciones a repetir
4 }
5 // Set de instrucciones que se ejecutaran luego de que la
   condicion de repeticion no se haya cumplido
```

Ejemplo while

```
1 // El programa imprime los numeros 10, 9, 8, ..., 1, 0
2 int i=10;
3
4 while (i >=0){
5     cout<<i<<endl;
6     i=i-1;
7 }
```

Contenido

- 1 Sentencia de Iteración: while
- 2 Sentencia de Iteración: do-while
- 3 Sentencia de Iteración: for
- 4 Sentencias Break y Continue

Sentencia de Iteración do-while

La sentencia de iteración do-while permite ejecutar un set instrucciones, **mientras** se cumpla una condición, es decir el set de instrucciones se ejecuta de forma repetitiva. A diferencia de **while** esta sentencia evalúa la condición al final de la repetición, este sutil pero importante cambio hace que do-while, sea útil en pedidos de datos y la generación de permutaciones.

```
1 do{  
2     // Set de instrucciones a repetir  
3 }while(condicion_de_repeticion);  
4 // La condicion_de_repeticion es una expresion booleana
```

Ejemplo do-while

```
1 //Este codigo pide la contrasenia a un usuario hasta que
   introduzca la contrasenia correcta que es CDSenior
2 do{
3     cout<<"Ingrese su contrasenia"<<endl;
4     cin>>contrasenia;
5 }while(contrasenia!="CDSenior");
```


Contenido

- 1 Sentencia de Iteración: while
- 2 Sentencia de Iteración: do-while
- 3 Sentencia de Iteración: for
- 4 Sentencias Break y Continue

Sentencia de Iteración for

La sentencia de iteración **for** permite ejecutar un set instrucciones un determinado número de repeticiones. Se suele decir que **for** es una forma abreviada y sintáctica de un **while**.

```
1 for(contador; condicion_de_finalizacion; incremento){  
2     // Set de instrucciones a repetir  
3 }
```

Ejemplo for

```
1 // For que imprime 1 2 3 ... 9 10
2 for(int i=1;i<=10;i++){
3     cout<<i<<" ";
4 }
5
6 // For que imprime 10 9 8 ... 2 1 0
7 for(int i=10;i>=0;i--){
8     cout<<i<<" ";
9 }
10
11 // For que imprime 4 7 10 13 ... 31
12 for(int i=4;i<=32;i=i+3){
13     cout<<i<<" ";
14 }
```

Contenido

- 1 Sentencia de Iteración: while
- 2 Sentencia de Iteración: do-while
- 3 Sentencia de Iteración: for
- 4 Sentencias Break y Continue

break

La sentencia **break** se utiliza para **cortar** o **romper** un ciclo iterativo. Sus usos más comunes son cortar ciclos cuando ya se ha encontrado una solución o un dato que satisfaga una condición. Cuando una sentencia **break** se ejecuta se rompe el ciclo y no se ejecutan las iteraciones restantes.

```
1 // Imprime 1 2 3 4
2 for( int i=1;i <=10;i++){
3     if( i==5) break;
4     cout<<i<<" ";
5 }
```

La sentencia **continue** se utiliza para **obviar** o **saltar** una iteración o caso. Sus usos más comunes son los de obviar ciertos casos para no ejecutar código innecesariamente. Cuando una sentencia **continue** se ejecuta se obvia todo el código restante de esa iteración.

```
1 // Imprime 1 2 3 4 6 7 9 10
2 for(int i=1;i<=10;i++){
3     if(i==5 or i==8) continue;
4     cout<<i<<" ";
5 }
```