Lenguaje de Programación

C++

Estructuras de control:**if**

José Luis Alonzo Velázquez

Universidad de Guanajuato

Sesión 23

Estructuras de selección

C++ tiene dos estructuras de control para la selección, **if** (selección simple y binaria) y **switch** (selección múltiple).

Sintaxis de la estructura de control **if**

if(⟨Condición⟩){
⟨Instrucción⟩
⟨Instrucción⟩
⋮
⟨Instrucción⟩
}else{
⟨Instrucción⟩
⟨Instrucción⟩
⋮
⟨Instrucción⟩
}

Ejemplo

```
#include <stdio.h>
using namespace std;
int main(){
    int numero;
    printf("Escribe un numero: ");
    scanf("%d",&numero);
    if(numero >= 4){
        printf("El numero  %d >= 4",numero);
    }else{
        printf("El numero  %d < 4",numero);
    }
    return 0;
}
```

Checar paridad de un número

```
#include <stdio.h>
using namespace std;
int main(){
    int numero;
    printf("Escribe un numero: ");
    scanf("%d",&numero);
    if((numero%2)==0){
        printf("El numero  %d es par",numero);
    }else{
        printf("El numero  %d es impar",numero);
    }
    return 0;
}
```

Error clásico

if(123 == a) ...
if(a == 123) ...

Error clásico

if(123 == a) ...
if(a == 123) ...

Si nos equivocamos al escribir estas expresiones, y ponemos sólo un signo "=", en el primer caso obtendremos un error del compilador, ya que estaremos intentando cambiar el valor de una constante, lo cual no es posible. En el segundo caso, el valor de la variable cambia, y además el resultado de evaluar la expresión no dependerá de una comparación, sino de una asignación, y siempre será "true", salvo que el valor asignado sea 0.

Funcionamiento de if, ¿qué sucede?

```
#include <stdio.h>
using namespace std;
int main(){
    int numero;
    printf("Escribe un numero: ");
    scanf("%d",&numero);
    if(numero=0){// siempre será "false"
        printf("El numero  %d es cero",numero);
    }else{
        printf("El numero  %d no es cero",numero);
    }
    return 0;
}
```

Funcionamiento de if, ¿qué sucede?

```
#include <stdio.h>
using namespace std;
int main(){
    int numero;
    printf("Escribe un numero: ");
    scanf("%d",&numero);
// siempre será "true", ya que 13 es distinto de 0
    if(numero=13){
        printf("El numero  %d es cero",numero);
    }else{
        printf("El numero  %d no es cero",numero);
    }
    return 0;
}
```

Un uso más

```
if (operador == +)
    resultado = A + B;
else if (operador == -)
    resultado = A - B;
else if (operador == *)
    resultado = A * B;
else if (operador == /)
    resultado = A / B;
else
    cout << "Operador invalido";
```

Tipos de Errores

Los compiladores clasifican los errores en dos tipos, dependiendo de lo serios que sean:

Tipos de Errores

Los compiladores clasifican los errores en dos tipos, dependiendo de lo serios que sean:

- "**Errores**": son errores que **impiden que el programa pueda ejecutarse**, los programas con "errores" no pueden pasar de la fase de compilación a la de enlazado, que es la fase en que se obtiene el programa ejecutable.

Tipos de Errores

Los compiladores clasifican los errores en dos tipos, dependiendo de lo serios que sean:

- "**Errores**": son errores que **impiden que el programa pueda ejecutarse**, los programas con "errores" no pueden pasar de la fase de compilación a la de enlazado, que es la fase en que se obtiene el programa ejecutable.
- "**Warnings**": son errores de poca entidad, (según el compilador que, por supuesto, no tiene ni idea de lo que intentamos hacer). **Estos errores no impiden** pasar a la fase de enlazado, y por lo tanto es posible ejecutarlos. Debes tener cuidado si tu compilador de da una lista de "warnings", eso significa que has cometido algún error, en cualquier caso repasa esta lista e intenta corregir los "warnings".

- Como Programar en C/C++, Deitel (Prentice Hall), 2da Edición.
- Programming Principles and Practice Using C++, Bjarne Stroustrup.
- http://www.codeblocks.org
- http://www.wxwidgets.org
- (O'Reilly) Practical C Programming (3rd Edition)