Lenguaje de Programación: C++ Estructuras de control:**FOR**

José Luis Alonzo Velázquez

Universidad de Guanajuato

Septiembre 2010

Estructuras de iteración

La estructura de control que veremos es la estructura de iteración **for**, la cual nos permite repetir un bloque de instrucciones un número definido de veces. Esta será muy parecida a la instrucción **iterate** que utilizábamos en Karel.

Sintaxis de la estructura de control **for**

```
for (<expres_ini>;<expres_bool>;<expres_inc>){
        <instruccion>
        <instruccion>
              ⋮
        <instruccion>
}
```

Inicialización de las variables

La parte de la inicialización (<expres_ini>), inicializa las variables de control del bucle. Se puede utilizar variables de control de bucle simples o múltiples. Lo más normal es inicializar en este punto una sola variable cuyo valor varía luego en la parte de incremento. Si se inicializan varias variables de control, cada inicialización se separa de la anterior con una coma.

Inicialización de las variables

La parte de la inicialización (<expres_ini>), inicializa las variables de control del bucle. Se puede utilizar variables de control de bucle simples o múltiples. Lo más normal es inicializar en este punto una sola variable cuyo valor varía luego en la parte de incremento. Si se inicializan varias variables de control, cada inicialización se separa de la anterior con una coma.

Expresión Lógica

Parte de iteración (<expres_bool>), que contiene una expresión lógica que hace que el bucle realice las iteraciones de las sentencias, mientras que la expresión sea verdadera.

Inicialización de las variables

La parte de la inicialización (<expres_ini>), inicializa las variables de control del bucle. Se puede utilizar variables de control de bucle simples o múltiples. Lo más normal es inicializar en este punto una sola variable cuyo valor varía luego en la parte de incremento. Si se inicializan varias variables de control, cada inicialización se separa de la anterior con una coma.

Expresión Lógica

Parte de iteración (<expres_bool>), que contiene una expresión lógica que hace que el bucle realice las iteraciones de las sentencias, mientras que la expresión sea verdadera.

Expresión de incremento

Parte de incremento (<expres_inc>), que modifica la variable o variables de control de bucle. Si se modifican varias variables de control, cada operación se separa de la anterior por una coma.

Ejemplo

```
#include <stdio.h>
using namespace std;
int main(){
    int n;
    for (n=0; n < 10 ; n++){
        printf("El valor de n es: %d\n",n);
    }
    return 0;
}
```

Ventaja de c++

```
#include <stdio.h>
using namespace std;
int main(){
    for (int n=0; n < 10 ; n++){
        printf("El valor de n es: %d\n",n);
    }
    return 0;
}
```

En este caso la definición de la variable **n** es solo temporal, y existirá solo mientras se realize el bucle **for**.

Equivalencia entre **for** y **while**

```
int n;
for (n=0; n < 10 ; n++){
    printf("El valor de n es: %d\n",n);
}
```

```
int n;
n=0
while(n < 10){
    printf("El valor de n es: %d\n",n);
    n++;
}
```

Consideraciones

- Debemos asegurarnos que la expresión de inicialización del bucle y la expresión de incremento harán que la condición del bucle se convierta en falsa en algún momento.

Consideraciones

- Debemos asegurarnos que la expresión de inicialización del bucle y la expresión de incremento harán que la condición del bucle se convierta en falsa en algún momento.
- Si el cuerpo de un bucle (secuencia de sentencias) modifica los valores de cualquiera de las variables implicadas en ese bucle, entonces el número de repeticiones se puede modificar.

¿Qué sucede en este código?

```
int limite = 1;
int i;
for (i=0; i<=limite; i++){
    printf("%d \n",i);
    limite++;
}
```

¿Qué sucede en este código?

```
int limite = 1;
int i;
for (i=0; i<=limite; i++){
    printf("%d \n",i);
    limite++;
}
```

Resultado

Producirá una secuencia infinita de enteros. **No** es una buena práctica de programación **modificar** el **valor** de la **variable de control**, por lo que evitaremos hacerlo.

Problema para clase

Escriba un programa que lea un número n e imprima la suma

$$1 + 2 + 3 + 4 + \cdots + (n-1) + n$$

.

Ejemplos

n=1;
la suma es: 1
n=5;
la suma es: 15
n=8;
la suma es: 36

 Programming Principles and Practice Using C++, Bjarne Stroustrup.

 http://www.codeblocks.org

 http://www.wxwidgets.org

 (O'Reilly) Practical C Programming (3rd Edition)