UNIVERSIDAD DE MURCIA

Fundamentos de Informática
6. Estructuras de control
(y sus algoritmos)

Fundamentos de Informática
Grado en Ingeniería Química

2

Contenidos

- *Iteraciones*
  - *while*
  - *for*
  - *do while*
  - *algunos usos de las iteraciones*
- *Condiciones*
  - *if*
  - *switch*

3

Estructuras de control

Iteraciones
(while)

4

Iteraciones (while)

```
#include <stdio.h>
main() {
    int contador;

    contador=0;

    while (contador<10)
    {
        printf("Número %d\n",contador);
        contador = contador+1;
    }

    getchar();
}
```

while: Se repite el bloque mientras se cumpla la condición.

while es una palabra reservada.

Iteraciones (while)

while (*condición*)
sentencia;

"sentencia;" puede ser una sola instrucción, o un bloque de instrucciones entre llaves

```
while (condición)
{
   sentencia;
   sentencia;
   …
}
```


Iteraciones (while)

```
contador=0;
while (contador<10)
{
     printf("Número %d\n",contador);
     contador=contador+1;
}
```

En bucles sencillos existe una *variable de control del bucle*.

La variable de control tiene que ser ***inicializada, comprobada y actualizada***.

En C la condicion puede ser cualquier expresión

```
a=b=x=2;
while ( (a*b)+2 != x+8)
{
     a=a+(c*2);
     x=x+1;
}
```

7

Estructuras de control

Iteraciones
(for)

8

Iteraciones (for)

```
contador=0;
while (contador < 5)
{
      printf("%d ",contador);
      contador = contador + 1;
}
```

Inicialización

Condición

Actualización

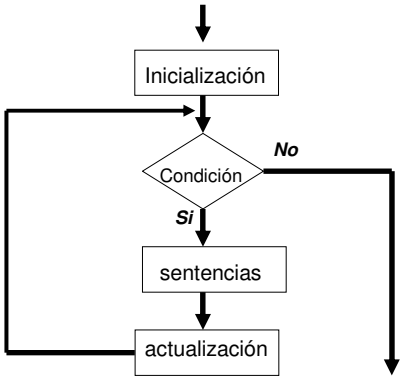
Inicialización *Condición* *Actualización*

```
for (contador = 0; contador < 5; contador ++)
{
      printf("%d ",contador);
}
```

9

Iteraciones (for)

for (*inicialización*; *condición*; *actualización*)
sentencia;



"sentencia;" puede ser una sola instrucción, o un bloque de instrucciones entre llaves

```
for (inicializac; condición; actualizac)
{
   sentencia;
   sentencia;
   …
}
```

10

Iteraciones (for)

```
multi=1;
sum=0;

for (i = 0; i < 5; i ++)
{
      multi=multi*2;
      sum=sum+2;
}

printf("%d %d",multi,sum);
```

for es muy útil para repetir un número conocido de veces

```
multi=1;
sum=0;
i=0;
while (i<5)
{
      multi=multi*2;
      sum=sum+2;
      i++;
}

printf("%d %d",multi,sum);
```

Iteraciones (for)

```
a = 3*x+b;
while ( (a*b)+2 != x+8)
{
      a=(c*2);
      x=x+1;
}
```

En C la inicialización, condición y actualización pueden ser expresiones complejas

```
for (a = 3*x+b; (a*b)+2 != x+8 ; a=c*2)
{
      x=x+1;
}
```


Iteraciones (for)

```
double p;
for (p=0.75; p<=5.5; p+=0.25)
      sentencia;
```

```
double p;
for (p=pow(y,3.0); p>2.0; p=sqrt(p))
      sentencia;
```

Instrucción mucho más potente que en otros lenguajes

```
for ( i=0 ; i<5 ; printf("i: %d\n",i++) ) ;
```

¡no hay sentencias!

13

Estructuras de control

Iteraciones
(do while)

14

Iteraciones (do while)

while

```
scanf("%d",&a);

while (a<10)
{
      printf("a: %d\n",a);
      a++;
}
```

do while

```
scanf("%d",&a);

do
{
      printf("a: %d\n",a);
      a++;
} while (a<10)
```

while *evalúa la condición al principio; puede entrar al bucle o no.*

do while *evalúa la condición al final; siempre entra al menos una vez.*

15

Estructuras de control

Algunos usos de las iteraciones

16

Iteraciones (centinelas)

Bucles controlados por ***centinelas***

Se puede usar un valor especial (centinela) como valor de finalización

Este programa suma los números que se introducien

Acaba cuando introduzca el valor -1

```
char centinela=-1; int valor, suma=0;

printf (“Introduce valor (-1 para acabar):”);
scanf(“%d”,&valor);

while (valor != centinela)
{
      suma=suma+valor;
      printf (“Introduce valor (-1 para acabar):”);
      scanf(“%d”,&valor);
}
printf(“Suma: %d\n”,suma);
```

17

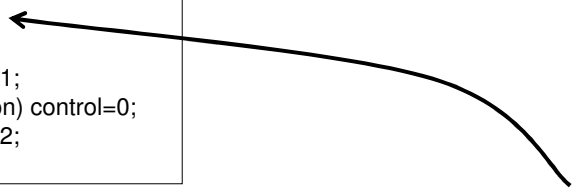
Iteraciones (banderas)

Bucles controlados por ***banderas (flag)***

Se establece una variable a 1, y cuando la iteración debe acabar, se pone a 0

```
char control=1;

while (control)
{
      sentencias1;
      if (expresión) control=0;
      sentencias2;
}
```



*Se repite indefinidamente, hasta que **control** se fije a 0*

18

Iteraciones (anidados)

Bucles anidados

```
for ( x = 1 ; x <= ultimox ; x++ )
      for (y = 1 ; y <= ultimoy ; y++ )
      {
            producto=x*y;
            printf("%d x %d = %d\n",x,y,producto);
      }
```

```
for ( x = 1 ; x <= ultimox ; x++ )
{
      producto=x*y;
      for (y = 1 ; y <= ultimoy ; y++ )
            printf("%d, ",y);
      printf("\n%d\n",x);
}
```

19

Estructuras de control

Condiciones
(switch)

20

Condiciones (switch)

switch: para múltiples alternativas. Especialmente buena cuando es una variable o expresión que puede tomar distintos valores.

switch, case, break y *default* son palabras reservadas de C.

```
scanf("%d",opcion);
switch (opcion)
{
      case 0:
            a=a+1; b=5;
            break;
      case 1:
            c=a+1;
            d=2;
            break;
      case 2:
            c=b-1; b=4;
            break;
      default:
            h=1; b=2;
}
```

Condiciones (switch)

El selector tiene que ser un int o char

```
switch (selector)
{
      case etiqueta1 : sentencia; sentencia; ... sentencia; break;
      case etiqueta2 : sentencia; sentencia; ... sentencia; break;
      ...
      case etiquetan : sentencia; sentencia; ... sentencia; break;
      default : sentencia; sentencia; ... sentencia;
}
```

default establece qué debe hacerse si no se da ningún caso.

break marca el final de las acciones de cada caso

Condiciones (switch)

break es opcional. Si no se pone, continúa ejecutando instrucciones hasta encontrar un break o el fin del switch.

```
printf("Mete 0 o 1:");
scanf("%d",&a);

switch (a)
{
  case 0:
      printf("Cero\n");
  case 1:
      printf("Uno\n");
      break;
  default:
      printf("Otro\n");
}
```

Como no hay break, si introduzco un 0, la salida será:

Cero
Uno

Si introduzco un uno, la salida será:

Uno

default es opcional, pero muy recomendable

Condiciones (switch)

Nos dice si el caracter es una vocal minúscula.

```
scanf("%c",caracter);
switch (caracter)
{
      case 'a': case 'e': case 'i':
      case 'o': case 'u':
            printf("vocal");
            break;
      default:
            printf("no vocal");
}
```

Cualquiera de los casos es vocal.

default en este caso es imprescindible.

Estructuras de control

- - - - - -