

# Bucles

# Tipos de bucle

- Número de repeticiones no conocido a priori:
  - while
  - do-while
- Número de repeticiones conocido a priori:
  - for

# Variable de control del bucle

1. Debe ser *inicializada* antes de comenzar el bucle
2. Su valor debe ser *evaluado/comprobado* en cada repetición del bucle
3. Su valor debe ser *actualizado* en cada repetición del bucle

# Bucles infinitos

- Cómo se detienen:
  - Ctrl-C
  - Linux: Matando el proceso (*kill -9*)
- Cómo se descubren:
  - Poniendo trazas en varios puntos del programa:  
`cout << "he llegado hasta aquí (4)" << endl;`
  - Utilizando el depurador

# Bucle `while`

- Sintaxis:

```
while (expression)  
statement
```

# Ejemplo 1

```
while (7)
    cout << "hello" << endl;
    cout << "good-bye" << endl;
```

## Ejemplo 2

```
short input;
cout << "enter number between 5 and 89, inclusive: ";
cin  >> input;

while (input < 5 || input > 89)
{
    cout << "that value is unacceptable... try again: ";
    cin  >> input;
}

cout << "the value " << input
      << "is in the interval [5, 89]" << endl;
```

# Ejemplo 3

```
long sum = 0;
short counter = 1;

while (counter <= 100)
{
    sum += counter;
    counter++ ;
}

cout << "sum of first 100 integers: " << sum;
```



# Ejemplo 4

```
const string QUESTION = "Can you guess my secret number? ";
const short ANSWER = 1234;
short ans;

cout << QUESTION;
cin >> ans;

while ( (ans - ANSWER) != 0 )
{
    cout << "Incorrect number. Please try again." << endl;
    cout << QUESTION;
}

cout << "You found it! Congratulations!" << endl;
```

# Bucle do-while

- Sintaxis:

**do**

**statement**

**while (expression) ;**

# Ejemplo 1

```
const short MAX_AGE = 116, MAX_ALLOWABLE_TRIES = 3;
short age, count = 0;
do
{
    cout << "Please enter your age: ";
    cin >> age;
    count++;

    if (age <= 0 || age > MAX_AGE)
    {
        cout << "This is not a valid value!" << endl;

        if (count == MAX_ALLOWABLE_TRIES)
            exit(1);
    }
} while (age <= 0 || age > MAX_AGE);
```

# Ejemplo 2

```
int number;  
do  
{  
    cout << "Enter an integer between 2 and 174"  
        << "that is a multiple of 3: ";  
    cin  >> number;  
} while ( /* ??? */ );
```

# Bucle for

- Sintaxis:

```
for (expression1; expression2; expression3)  
    statement
```

# Ejemplo 1

```
short sum;  
  
cout << "What is 2+2? ";  
cin  >> sum;  
  
for (; sum != 4;)  
{  
    cout << "Incorrect!! What is 2+2? " << endl;  
    cin  >> sum;  
}
```

## Ejemplo 2

```
float average;  
long max, sum = 0;  
  
cout << "enter a positive integer: ";  
cin  >> max;  
  
for(int i = 0; i <= max; i++)  
    sum += i;  
  
average = static_cast<float>(sum) / max;  
cout << "average is " << average << endl;
```

# Ámbito (*scope*) de variable de bucle

```
int i = 20;
```

```
for (int i = 1; i <= 10; i++)  
{  
    cout << "hello" << endl;  
}
```

```
cout << i << endl;
```



# Ejercicio

- Escribir un programa en C que imprima todos los años no bisiestos entre 1900 y 2000, incluyendo ambos en su caso.

# Bucles anidados

- Queremos imprimir lo siguiente:

```
* * * * *  
* * * *  
* * *  
* *  
*
```

# Bucles anidados

- Imprimimos 5 líneas:

```
for (int i = 0; i < 5; i++)  
{  
    cout << endl;  
}
```

# Bucles anidados

- Imprimimos 5 asteriscos:

```
for (int j = 0; j < 5; j++)  
{  
    cout << "* ";  
}
```

# Bucles anidados

- Combinamos (anidamos):

```
for (int i = 0; i < 5; i++)  
{  
    for (int j = 0; j < 5; j++)  
    {  
        cout << "* ";  
    }  
    cout << endl;  
}
```

# Bucles anidados

- Salida real vs. salida esperada:

	Objetivo					Actual				
i=1	*	*	*	*	*	*	*	*	*	*
i=2	*	*	*	*	*	*	*	*	*	*
i=3	*	*	*	*	*	*	*	*	*	*
i=4	*	*	*	*	*	*	*	*	*	*
i=5	*	*	*	*	*	*	*	*	*	*

# Bucles anidados

- Corregimos:

```
for (int i = 0; i < 5; i++)  
{  
    for (int j = 0; j < 5 - i; j++)  
    {  
        cout << "* ";  
    }  
    cout << endl;  
}
```

# Bucles anidados

- **Ejercicio:** imprimir lo siguiente partiendo de lo anterior:

```
* * * * *
 * * * *
  * * *
   * *
    *
```