

# Algoritmos y Estructura de Datos

Clase 3



**UTN.BA**

UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# Estructuras de Repetición

(Bucles/Loop)



**UTN.BA**

UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES

# ¿Para qué sirve?

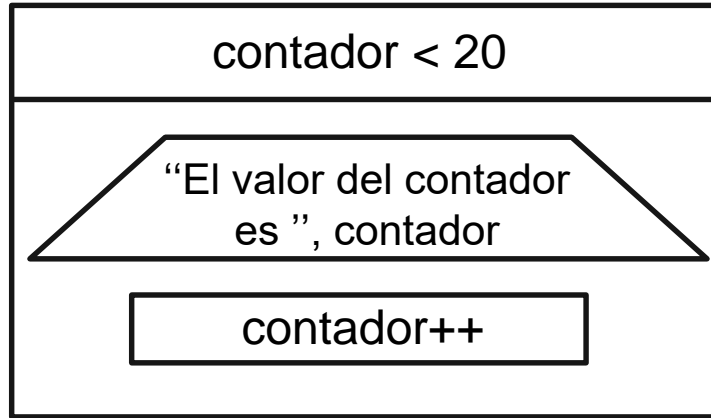
Una instrucción de repetición nos permite ejecutar un grupo de instrucciones varias veces.

¿Algún ejemplo de la vida cotidiana?

# while

En diagrama de Lindsay:

```
int contador = 10;
```



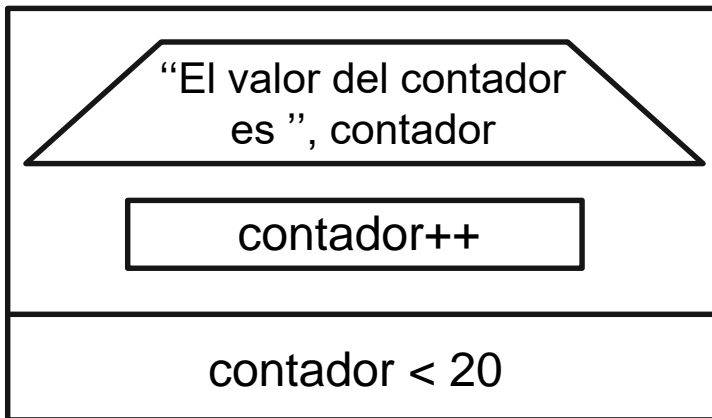
En C++:

```
while ( contador < 20 ) {  
    cout << "El valor del contador es: " << contador << endl;  
    contador++;  
}
```

# do...while

En diagrama de Lindsay:

```
int contador = 10;
```



En C++:

```
do {  
    cout << "El valor del contador es: " << contador << endl;  
    contador++;  
} while ( contador < 20 );
```



# for

Permite escribir eficientemente un bucle que necesita ejecutarse un número específico de veces.

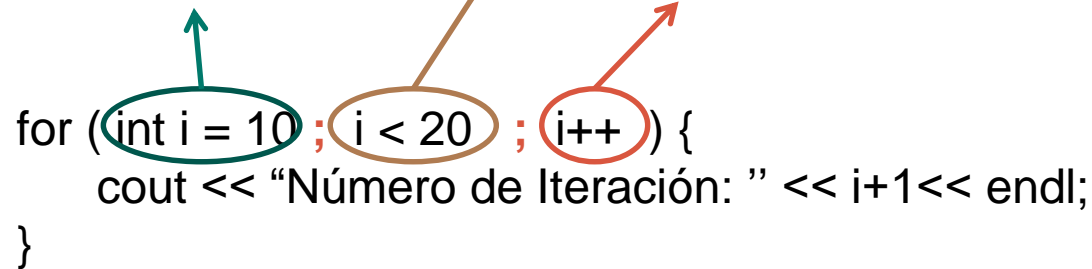
En C++:

declarar e inicializar  
cualquier variable de  
control del bucle

condición

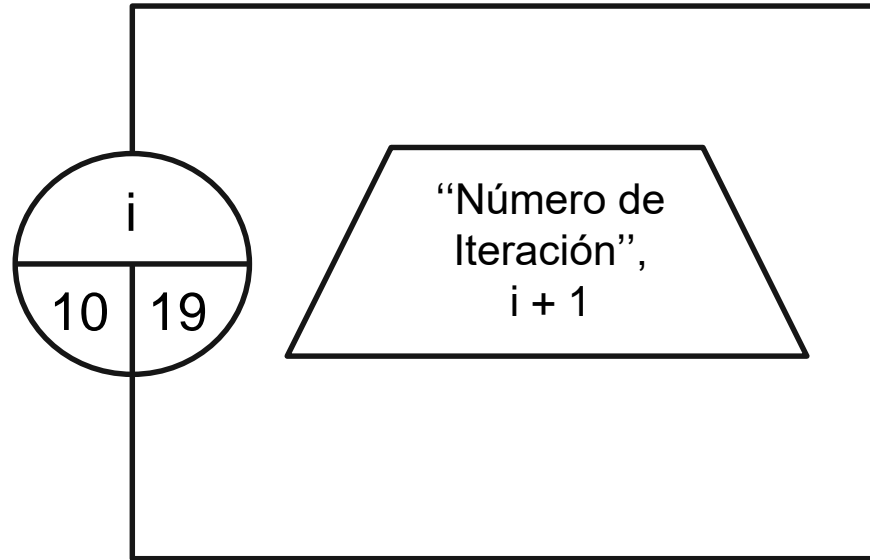
incremento

```
for (int i = 10 ; i < 20 ; i++) {  
    cout << "Número de Iteración: " << i+1 << endl;  
}
```



# for

En diagrama de Lindsay:



# El Loop infinito

Un bucle se vuelve infinito si la condición nunca se vuelve falsa.

```
for( ; ; ) {  
    cout << "Este bucle va a correr para siempre" << endl;  
}
```

¿Se les ocurre otro ejemplo?



# Resumen

Estructura	Cantidad de Repeticiones
while	0 a N
do...while	1 a N
for	N




# Estructura de un Programa en C++



**UTN.BA**

UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES



```
#include <iostream>
```

Línea especial interpretada  
antes de que comience la  
compilación del programa  
en sí.

```
int main() {
```

```
    std::cout << "Hello World!";
```

```
    return 0;
```

```
}
```

```
#include <iostream>
```

Línea especial interpretada antes de que comience la compilación del programa en sí.

```
int main(){
```

```
    std::cout << "Hello World!";
```

```
    return 0;
```

```
}
```

Es la función que se llama cuando se ejecuta el programa.

```
#include <iostream>
```

Línea especial interpretada antes de que comience la compilación del programa en sí.

```
int main() {
```

Es la función que se llama cuando se ejecuta el programa.

Es la salida estándar

```
std::cout << "Hello World!";
```

```
return 0;
```

```
}
```

cout es parte de la biblioteca estándar. Se declara dentro de lo que se denomina el espacio de nombres std

```
#include <iostream>
```

Línea especial interpretada antes de que comience la compilación del programa en sí.

```
int main(){
```

Es la función que se llama cuando se ejecuta el programa.

```
std::cout << "Hello World!";
```


```
return 0;
```

```
}
```

```
# include <iostream>  
using namespace std;
```

```
int main() {
```

Ya no hace falta  
usar std::



```
cout << "Hello World!";
```

```
    return 0;
```

```
}
```

# Comentarios



**UTN.BA**

UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES



Son **porciones de código ignoradas por el compilador** que permiten al usuario hacer **notas** en áreas relevantes del **código fuente**. Los comentarios vienen en forma de bloque o como líneas simples.

// Comentarios de una línea

/\* Comentarios

multilínea \*/



**UTN.BA**

UNIVERSIDAD TECNOLÓGICA NACIONAL  
FACULTAD REGIONAL BUENOS AIRES