


# Laboratorio de Computación I

Ciclo inexacto

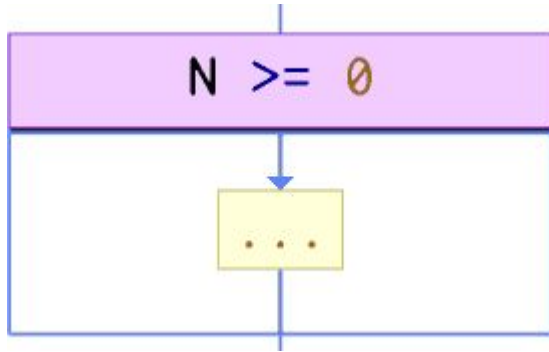
# Estructura de repetición

Las estructuras de repetición nos permiten ejecutar un conjunto de instrucciones una serie de veces. Se clasifican en **exacto** e **inexacto**.



# Ciclo inexacto

El ciclo inexacto nos permite ejecutar un conjunto de instrucciones una indeterminada cantidad de veces mientras se cumpla una o varias condiciones.



Diagrama

```
while (n >= 0){  
    /* Instrucciones a  
       repetir si n es >= 0*/  
}
```

Código C

# Ciclo inexacto - while

## while (condición)

La **condición** determina la continuidad del ciclo inexacto. Puede estar compuesta por una o varias proposiciones lógicas.

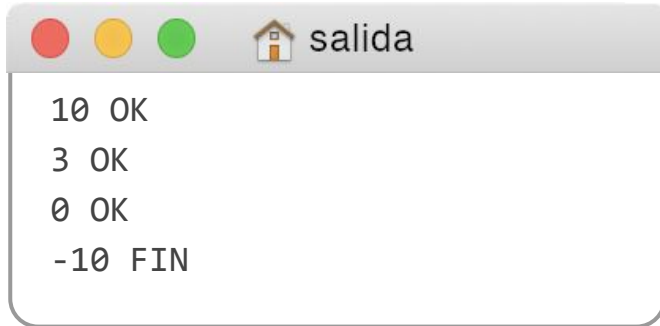
El ciclo iterará si la condición es **verdadera** y finalizará si la condición es falsa.

Es necesario garantizar que la condición pueda ser falsa en algún momento para no terminar en un **ciclo infinito**.

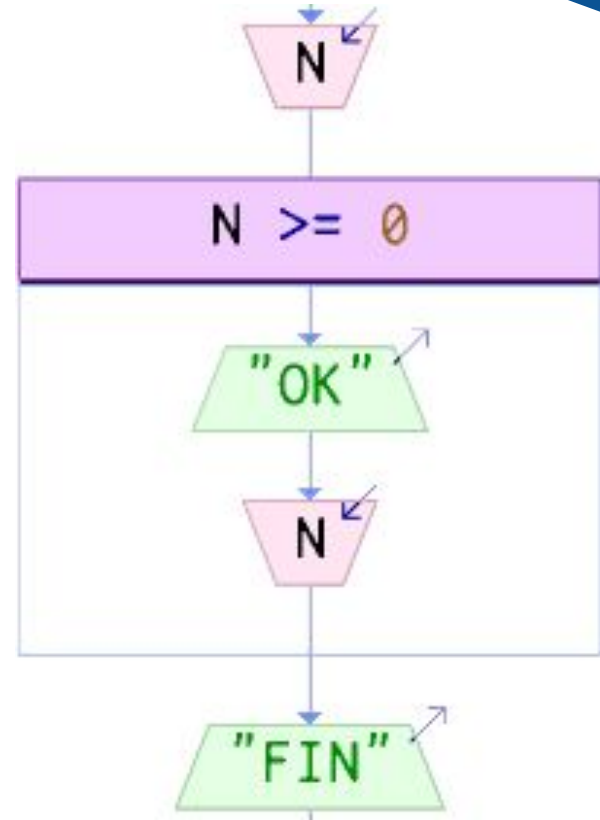
# Ejemplo

```
int n;  
cin >> n;  
while(n >= 0){  
    cout << "OK" << endl;  
    cin >> n;  
}  
cout << "FIN" << endl;
```

Código C/C++



Salida por consola



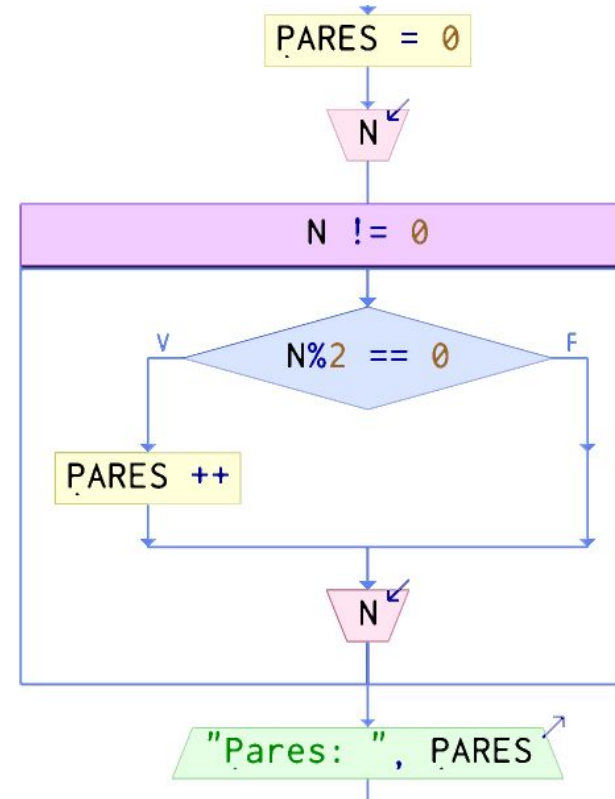
Diagrama

# Ejemplo

```
#include <iostream>
using namespace std;
int main(){
    int i, n, pares = 0;
    cin >> n;
    while(n != 0){
        if (n%2==0){
            pares++;
        }

        cin >> n;
    }
    cout << "Pares: ";
    cout << pares;
    return 0;
}
```

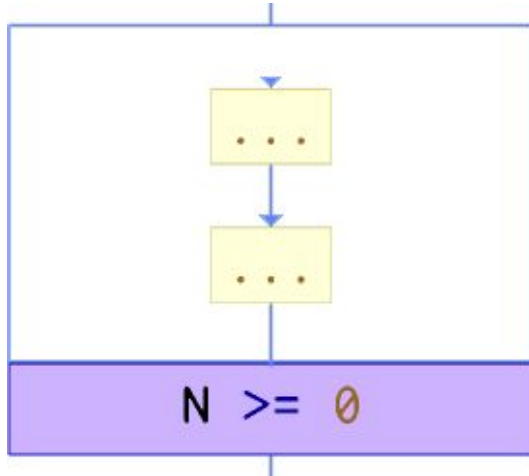
Código C/C++



Diagrama

# Ciclo inexacto - do while

La estructura **do-while** funciona de la misma manera que un ciclo while con una variante. Como su condición se evalúa al final, garantiza al menos una iteración.



Diagrama

```
do{  
    /* Instrucciones a  
       repetir si n es >= 0.  
       Esto se ejecutará al  
       menos una vez.  
    */  
}while (n >= 0);
```

Código C

# Ejercicios

<https://bit.ly/LAB1-TP04>