



# Loops

## Practica

Universidad Católica Boliviana

MSc, José Jesús Cabrera Pantoja

# Outline

- Recap
- Practica Loop

# Control de flujo

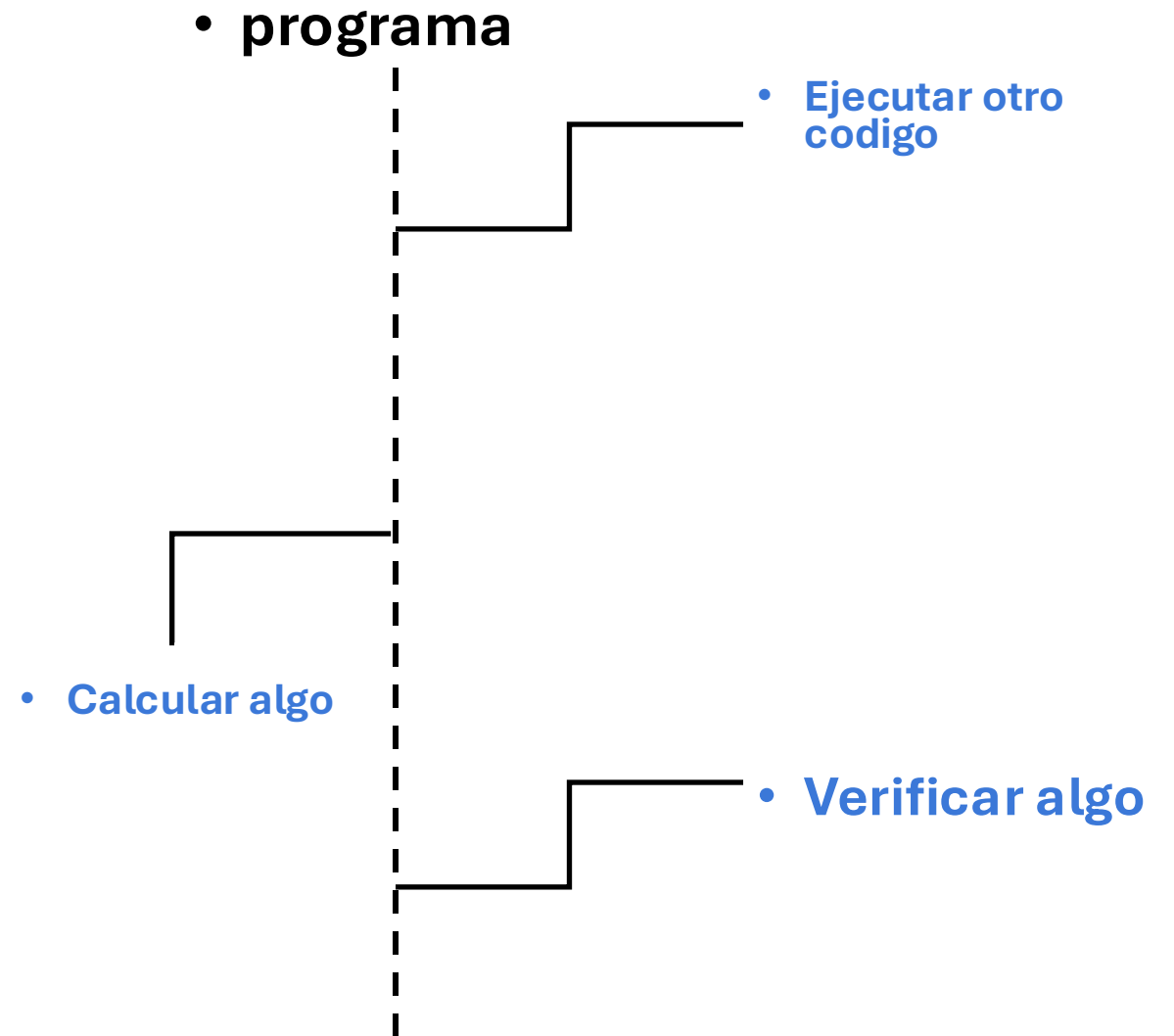
- Comienza por el principio, dijo el rey, muy gravemente, y continúa hasta que llegues al final: Entonces detente. **“Lewis Carroll, El maravilloso mundo de Alicia”**

- programa



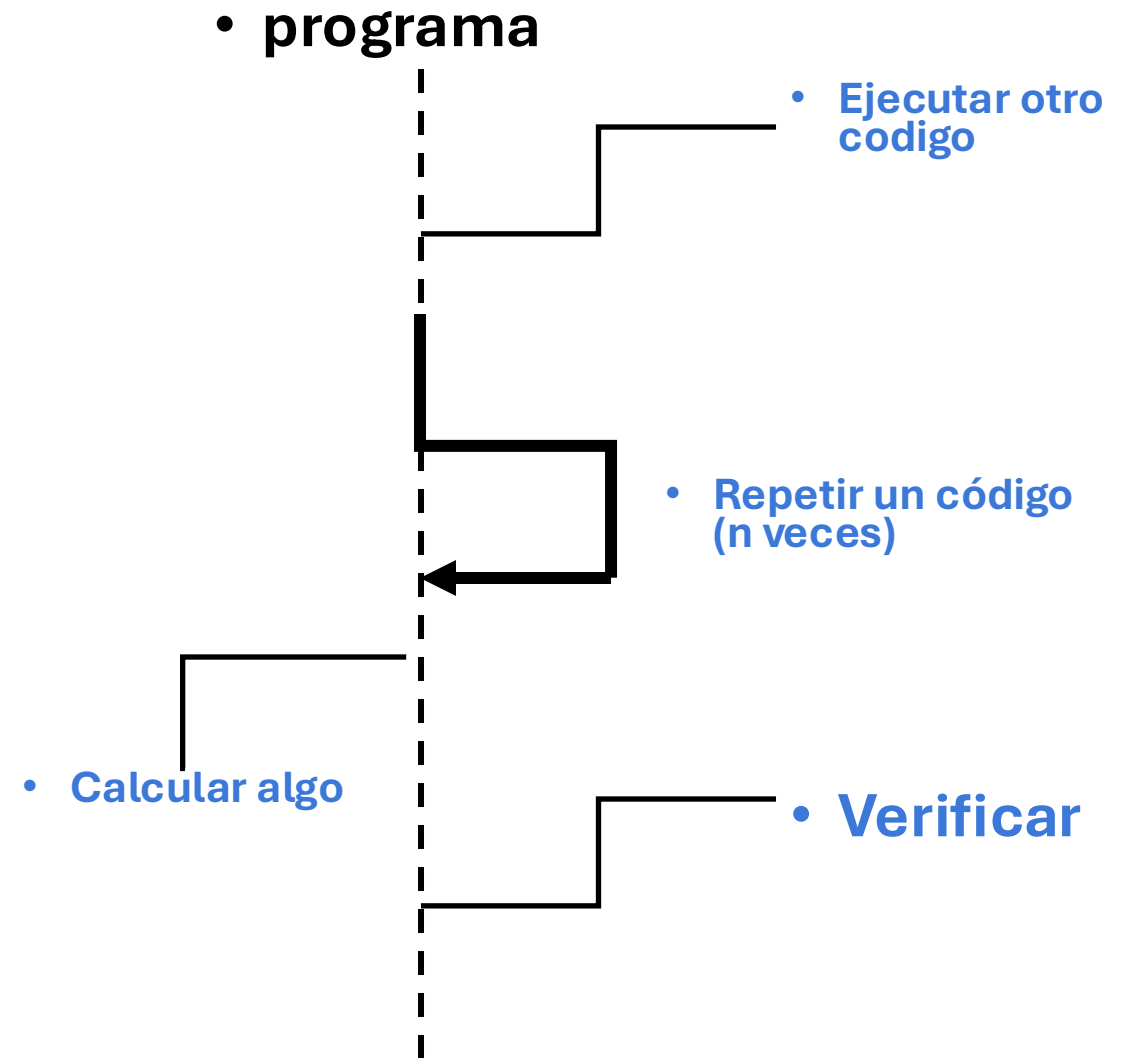
# Control de flujo

- Aquí es donde entran en juego las declaraciones condicionales como **IF**, **IF-ELSE** y **IF-ELSE-IF**.



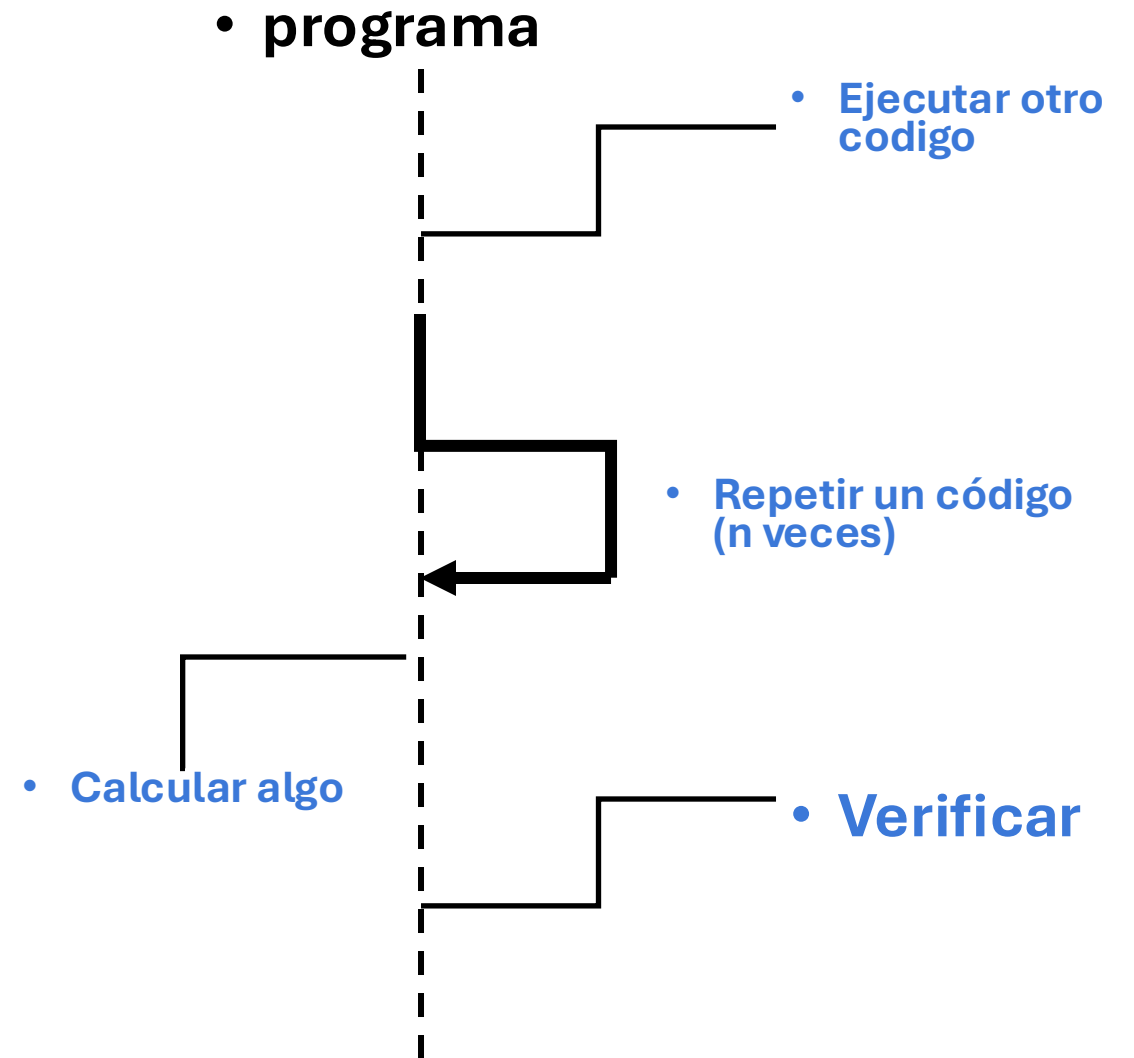
# Loop

- En los programas de computadora, una secuencia de instrucciones que se ejecuta hasta que una condición se vuelve falsa se llama **bucle (loop)**.




# Loops

- While loop
- For loop




# Loop: Implementacion

PseudoCodigo



```
1 i = 1
2 WHILE i <= 10
3     OUTPUT i
4     i = i + 1
```



```
1 while (haya helado en la heladera) {
2     ir y volver a la heladera por helado;
3 }
```

# Loop: Implementacion

PseudoCodigo



```
1 i = 1
2 WHILE i <= 10
3     OUTPUT i
4     i = i + 1
```

C++




```
1 int main() {
2     int i = 0;
3     while(i <= 10) {
4         cout << i << endl;
5         i = i + 1;
6     }
7
8     return 0;
9 }
```



# Loop: Implementacion

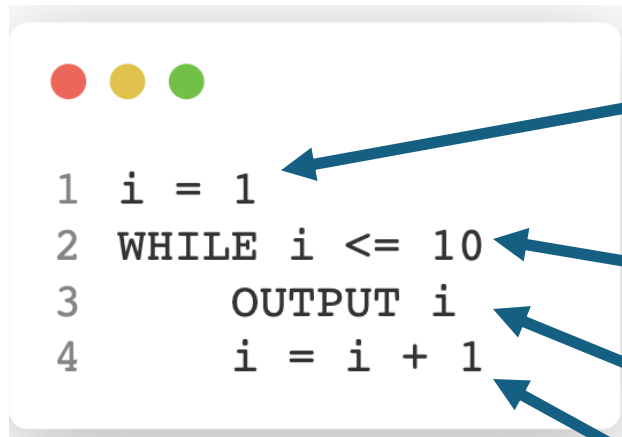
C++



```
1  int main() {  
2      int i = 0;  
3      while(i <= 10) {  
4          cout << i << endl;  
5          i += 1;  
6      }  
7  
8      return 0;  
9  }
```

# Loop: Implementacion

PseudoCodigo



```
1 i = 1
2 WHILE i <= 10
3     OUTPUT i
4     i = i + 1
```

- En la primera línea, establecemos  $i$  en 1.  $i$  contiene el número que mostramos.
- Luego, verificamos si  $i$  es menor o igual a 10.
- Si lo es, mostramos el valor de  $i$
- Lo incrementamos en 1.
- Repetimos estos pasos hasta que  $i$  sea mayor que 10.

# Loop: For



```
1 cantidad_de_pasos = 0;  
2 while (cantidad_de_pasos != 3) {  
3     ir a la heladera;  
4     cantidad_de_pasos = cantidad_de_pasos + 1;  
5 }
```

# Loop: For



```
1 cantidad_de_pasos = 0;
2 while (cantidad_de_pasos != 3) {
3     ir a la heladera;
4     cantidad_de_pasos = cantidad_de_pasos + 1;
5 }
```



```
1 for (<accion inicial>; <condicion>; <paso del ciclo>) {
2     <accion, la cual es necesaria repetir>
3 }
```

# Loop: For




```
1 cantidad_de_pasos = 0;
2 while (cantidad_de_pasos != 3) {
3     ir a la heladera;
4     cantidad_de_pasos = cantidad_de_pasos + 1;
5 }
```



```
1 for (cantidad_de_pasos = 0; cantidad_de_pasos != 3; cantidad_de_pasos += 1) {
2     ir a la heladera;
3 }
```

# Loop: concisión

- A menudo, en la escritura de pasos de ciclo se utilizan versiones cortas para **`i += 1`**. Hay dos opciones de este tipo:
  - **`++i`** - incremento de **prefijo**.
  - **`i++`** - incremento de **postfijo**.



```
1 int main() {  
2  
3     for (int i = 0; i != 3; i++) {  
4         cout << "Check the fridge"s << endl;  
5     }  
6  
7     return 0;  
8 }
```

# Loop: Problema

- **Suponga que tiene un entero positivo  $n$  que es mayor que uno y desea saber si es primo.**

# Loop: Problema

- **Entender el problema**
- Suponga que tiene un entero positivo  $n$  que es mayor que uno y desea saber si es primo.



# Loop: Problema

- **Entender el problema**
- Suponga que tiene un entero positivo  $n$  que es mayor que uno y desea saber si es primo.
- Entradas
- Salidas

# Loop: Problema

- **Encontrar una posible solución**
- Recuerde la definición de un número primo.
- Un número primo no es divisible por ningún otro número excepto por sí mismo y por 1.
- Por lo tanto, una forma posible de verificar si  $n$  es primo es asegurarse de que no sea divisible por ningún número desde 2 hasta  $n / 2$
- si  $n / 2$  es un número decimal, luego redondearlo hacia abajo al entero más cercano

# Loop: Problema


- **Probar la posible solucion**
- Por ejemplo, para comprobar si 11 es un número primo, lo dividimos por 2, 3, 4 y 5 y vemos si el resto es cero.
- Note que ninguno de los números dados (2, 3, 4 y 5) divide exactamente a 11, es un número primo que tiene factores solo 1 y 11

# Loop: Problema

- Ejecutar la solución en diferentes casos de prueba.
- Por ejemplo, 5 es primo y el método anterior da la respuesta correcta.

# Loop: Problema

- Pseudo codigo



```
1 INPUT n
2 i = 2
3 answer = prime
4 WHILE i <= n / 2
5     rem = n % i
6     IF rem is not equal to 0
7         i = i + 1
8     ELSE
9         answer = not prime
10    END WHILE LOOP
11 OUTPUT answer
```