

---

# Ejercicios de Programación: Bucles

---

## Módulo 1: Simulación Bancaria

### Ejercicio 1: Calculadora de Interés Compuesto (`for`)

**Escenario:** Trabajas para un banco y necesitas crear una herramienta que permita a los clientes proyectar sus ahorros. La herramienta debe calcular cómo crecerá una inversión inicial a lo largo de los años gracias al interés compuesto.

**Tu tarea:**

1. **Solicita al usuario tres datos:**
  - La **cantidad inicial de dinero** a invertir (ej: 1000€).
  - La **tasa de interés anual** en porcentaje (ej: 5%).
  - El **número de años** que mantendrá la inversión (ej: 10 años).
2. Usa un **bucle `for`** que se ejecute una vez por cada año de la inversión.
3. En cada iteración, calcula el interés ganado ese año y súmalo al total. La fórmula para el interés de un año es: `saldo_actual * (tasa_de_interes / 100)`.
4. Muestra por pantalla un resumen anual, indicando el año y el saldo total al final de ese año.

**Ejemplo de salida:**

```
Año 1: Saldo de 1050.00€
Año 2: Saldo de 1102.50€
...
Año 10: Saldo de 1628.89€
```

---

### Ejercicio 2: El Cajero Automático Interminable (`while`)

**Escenario:** Estás programando el software de un cajero automático. El cajero debe permitir al usuario retirar dinero mientras tenga saldo suficiente. El programa no sabe cuántas veces el usuario querrá retirar dinero, por lo que debe continuar hasta que el usuario decida salir o se quede sin fondos.

**Tu tarea:**

1. **Pregunta al usuario su saldo inicial** (ej: 2000€).
2. Crea un **bucle `while`** que se ejecute mientras el saldo sea mayor que cero y el usuario no haya decidido salir.
3. Dentro del bucle, pregunta al usuario **cuánto dinero desea retirar**.
4. **Comprueba** si tiene saldo suficiente.
  - Si puede retirarlo, resta la cantidad al saldo y muestra el nuevo saldo.

- Si no puede, muéstrale un mensaje de "Saldo insuficiente".
  - 5. Pregúntale si desea **realizar otra operación** (puedes usar un '1' para sí y un '0' para no). El bucle debe terminar si el usuario introduce un '0'.
- 

### Ejercicio 3: Validación de PIN de Seguridad (`do-while`)

**Escenario:** Antes de acceder a su cuenta online, un cliente debe introducir su PIN de 4 dígitos. El sistema debe pedir el PIN al menos una vez y seguir pidiéndolo hasta que el usuario introduzca el PIN correcto.

**Tu tarea:**

1. Define un **PIN correcto** en una variable dentro de tu programa (ej: 1234).
  2. Utiliza un **bucle `do-while`**. Este tipo de bucle es perfecto porque garantiza que el cuerpo del bucle (la pregunta del PIN) se ejecute **al menos una vez**.
  3. Dentro del bucle, **pide al usuario que introduzca su PIN**.
  4. El bucle debe repetirse **mientras** el PIN introducido sea **diferente** al PIN correcto.
  5. Cuando el usuario introduzca el PIN correcto, el bucle terminará y deberás mostrar un mensaje de "Acceso concedido".
- 

## Módulo 2: Gestión de una Tienda Online

### Ejercicio 4: Carrito de la Compra (`for`)

**Escenario:** Eres el desarrollador de una plataforma de e-commerce. Necesitas implementar la funcionalidad del carrito de la compra que calcule el precio total de una serie de productos. El usuario primero indicará cuántos productos va a comprar.

**Tu tarea:**

1. Primero, pregunta al usuario **cuántos productos** va a introducir en el carrito.
  2. Usa un **bucle `for`** que se repita exactamente ese número de veces.
  3. En cada iteración del bucle, pide al usuario el **precio del producto** actual.
  4. Suma ese precio a una variable que almacene el **total de la compra**.
  5. Al final del bucle, muestra el **precio total** que el cliente debe pagar.
- 

### Ejercicio 5: Sistema de Valoración de Productos (`while`)

**Escenario:** Tu web necesita un sistema para que los usuarios valoren un producto. Los usuarios pueden añadir una puntuación de 1 a 5 estrellas. El programa debe recoger todas las puntuaciones que los usuarios quieran introducir y finalmente calcular la valoración media.

### Tu tarea:

1. Informa al usuario que puede introducir puntuaciones (de 1 a 5) y que para terminar debe introducir un **cero (0)**.
  2. Usa un **bucle while** que se ejecute mientras la entrada del usuario no sea 0.
  3. Dentro del bucle:
    - Pide una **puntuación**.
    - Si la puntuación está entre 1 y 5, súmala a un total y aumenta un contador de cuántas valoraciones has recibido.
    - Si es otro número (que no sea 0), muestra un mensaje de error.
  4. Cuando el bucle termine (el usuario introduzca 0), calcula la **puntuación media** (total de puntos / número de valoraciones).
  5. Muestra la puntuación media final. ¡Cuidado con la división por cero si no se introduce ninguna valoración!
- 

## Ejercicio 6: Menú de Opciones de Usuario (**do-while**)

**Escenario:** Estás creando el panel de control de un usuario en una página web. Al entrar, el usuario ve un menú de opciones y debe elegir una. El menú debe mostrarse al menos una vez y seguir apareciendo hasta que elija la opción de "Salir".

### Tu tarea:

1. Usa un **bucle do-while** para mostrar el menú y pedir una opción.
  2. Dentro del bucle, muestra las siguientes opciones al usuario:
  3. Menú de Usuario:
    4. 1. Ver mi perfil
    5. 2. Modificar dirección de envío
    6. 3. Ver historial de pedidos
    7. 4. Salir
  8. Pide al usuario que **elija una opción** (introduciendo un número del 1 al 4).
  9. Usa una estructura de control (como **if-else** o **switch**) para mostrar un mensaje diferente según la opción elegida (ej: "Cargando tu perfil...").
  10. El bucle debe continuar repitiéndose **mientras** la opción elegida por el usuario no sea la **4 (Salir)**.
- 

## Módulo 3: Control de un Robot Sencillo

### Ejercicio 7: Secuencia de Movimientos del Brazo Robótico (**for**)

**Escenario:** Estás programando un brazo robótico industrial que debe realizar una secuencia de movimientos precisos para ensamblar una pieza. El operario primero indica cuántos pasos tiene la secuencia.

### Tu tarea:

1. Pregunta al operario **cuántos movimientos** va a programar en la secuencia.
  2. Usa un **bucle `for`** para que se repita ese número de veces.
  3. En cada iteración, pide las **coordenadas (X, Y)** a las que el robot debe moverse (puedes pedir los dos números por separado).
  4. Dentro del bucle, muestra un mensaje simulando la acción. Por ejemplo:  
Movimiento 1: Brazo robótico moviéndose a la posición (X, Y).
  5. Al final, muestra un mensaje de "Secuencia de movimientos completada".
- 

## Ejercicio 8: Robot Explorador en Bucle (`while`)

**Escenario:** Un pequeño robot explorador está en un planeta desconocido. El robot avanza y, tras cada avance, analiza el terreno que tiene delante. Debe continuar explorando hasta que su sensor detecte una "base segura".

**Tu tarea:**

1. Informa al usuario que puede introducir lecturas del sensor del robot (ej: "rocas", "cráter", "terreno llano", "base segura").
  2. Crea un **bucle `while`** que se ejecute mientras la lectura del sensor **no sea** "base segura".
  3. Dentro del bucle, pide al usuario la **nueva lectura del sensor**.
  4. Imprime un mensaje que indique lo que el robot está haciendo. Por ejemplo:  
Sensor detecta 'rocas'. El robot continúa explorando....
  5. Cuando el usuario finalmente introduce "base segura", el bucle termina y el programa debe imprimir: "¡Base segura encontrada! Misión cumplida."
- 

## Ejercicio 9: Diagnóstico del Sistema del Robot (`do-while`)

**Escenario:** Un robot de mantenimiento debe realizar un chequeo de sus sistemas internos al arrancar. El chequeo se ejecuta al menos una vez, y después pregunta al técnico si desea volver a ejecutarlo.

**Tu tarea:**

1. Usa un **bucle `do-while`** para simular el diagnóstico.
2. Dentro del bucle, muestra una serie de mensajes que simulen el chequeo:
3. Iniciando diagnóstico...
4. - Verificando sensores... OK
5. - Calibrando motores... OK
6. - Estado de la batería: 98%
7. Diagnóstico completado.
8. Después de mostrar los mensajes, pregunta al técnico: "**¿Desea ejecutar el diagnóstico de nuevo? (s/n)**".
9. El bucle debe repetirse **mientras** el usuario introduzca 's' (o 'S'). Si introduce cualquier otra cosa, el bucle terminará.
10. Al salir del bucle, muestra un mensaje como "Sistema listo. Robot preparado para operar."

