Clase 5 – Ciclos While y Automatización de Procesos

Marco Teórico

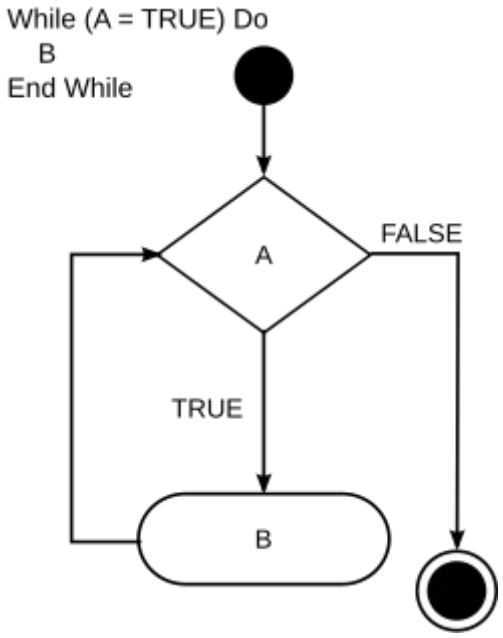
Los ciclos son estructuras de control que permiten repetir una serie de instrucciones varias veces. Son muy útiles cuando se necesita ejecutar una tarea muchas veces, y en lugar de escribir el mismo código una y otra vez, se puede utilizar un ciclo para hacerlo de forma más eficiente.

Existen diferentes tipos de ciclos, pero todos ellos se basan en una condición que se evalúa antes de cada iteración. Si la condición se cumple, se ejecuta el bloque de código que forma parte del ciclo; de lo contrario, el ciclo se detiene y se continúa con la ejecución del resto del programa.

El ciclo while ejecuta un bloque de instrucciones mientras una condición sea verdadera.

Estructura General

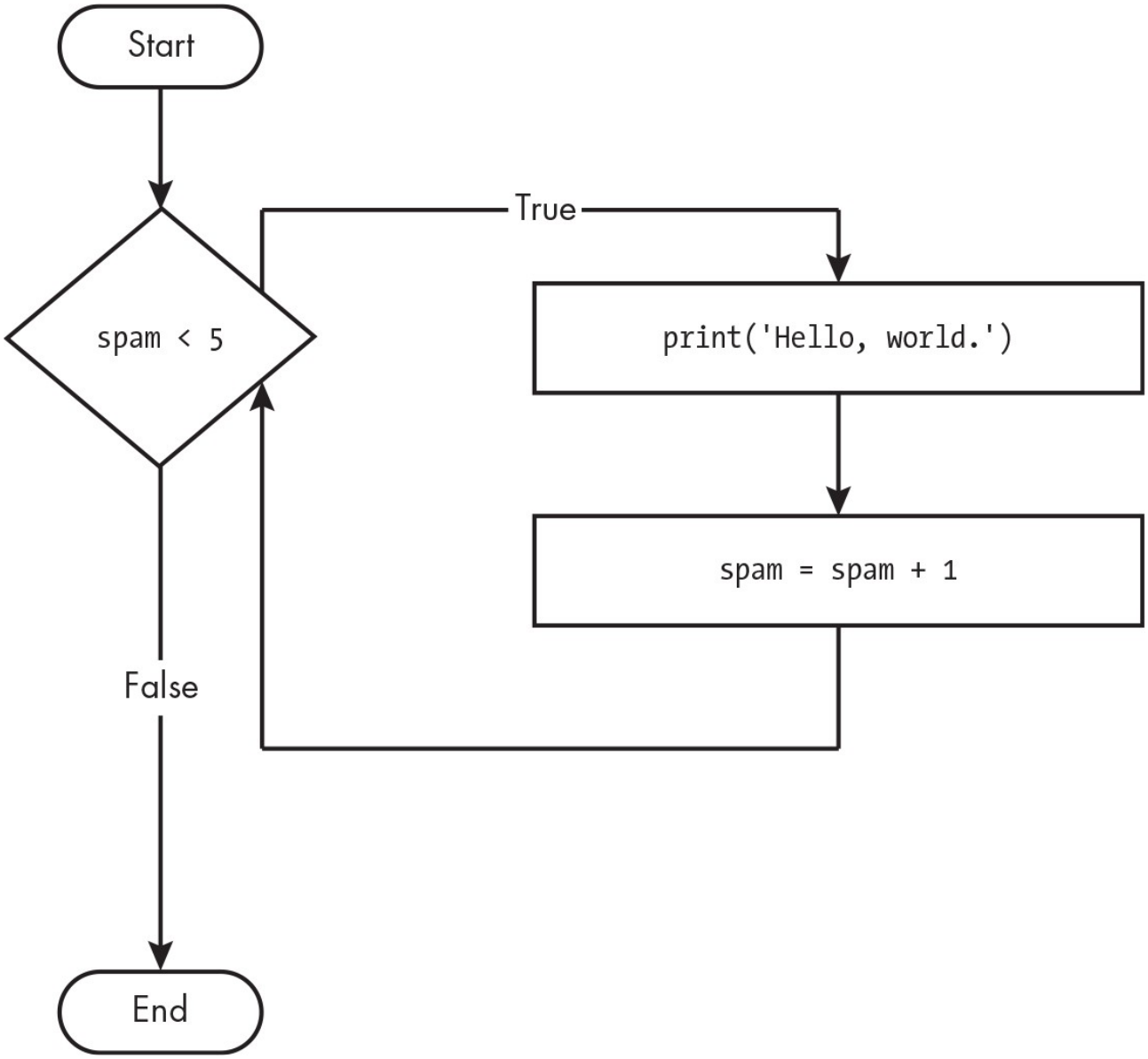
While condicion:



Conceptos centrales

- Condición lógica.
- Variable de control.
- Repetición controlada.
- Riesgo de bucles infinitos.
- Automatización de tareas.

```
contador = 1
while contador <= 5:
    print(contador)
    contador += 1
```



En el bucle while, la condición siempre se comprueba al comienzo de cada *iteración* (es decir, cada vez que se ejecuta el bucle). Si la condición es verdadera, entonces se ejecuta la cláusula, y después, la condición se comprueba de nuevo. La primera vez que se encuentra la condición es falsa, se omite la cláusula while.

Actividades Prácticas

Actividad 1 – Contador automático

Crear un programa que muestre números del 1 al 20.

Actividad 2 – Validación de contraseña

Solicitar una contraseña hasta que el usuario ingrese la correcta.

```
clave = ""

while clave != "python123":
    clave = input("Ingrese la contraseña: ")

print("Acceso permitido")
```

Actividad 3 – Cajero automático simple

Simular un sistema que permita retirar dinero mientras exista saldo disponible.

Actividad Integradora

Diseñar un menú repetitivo con opciones:

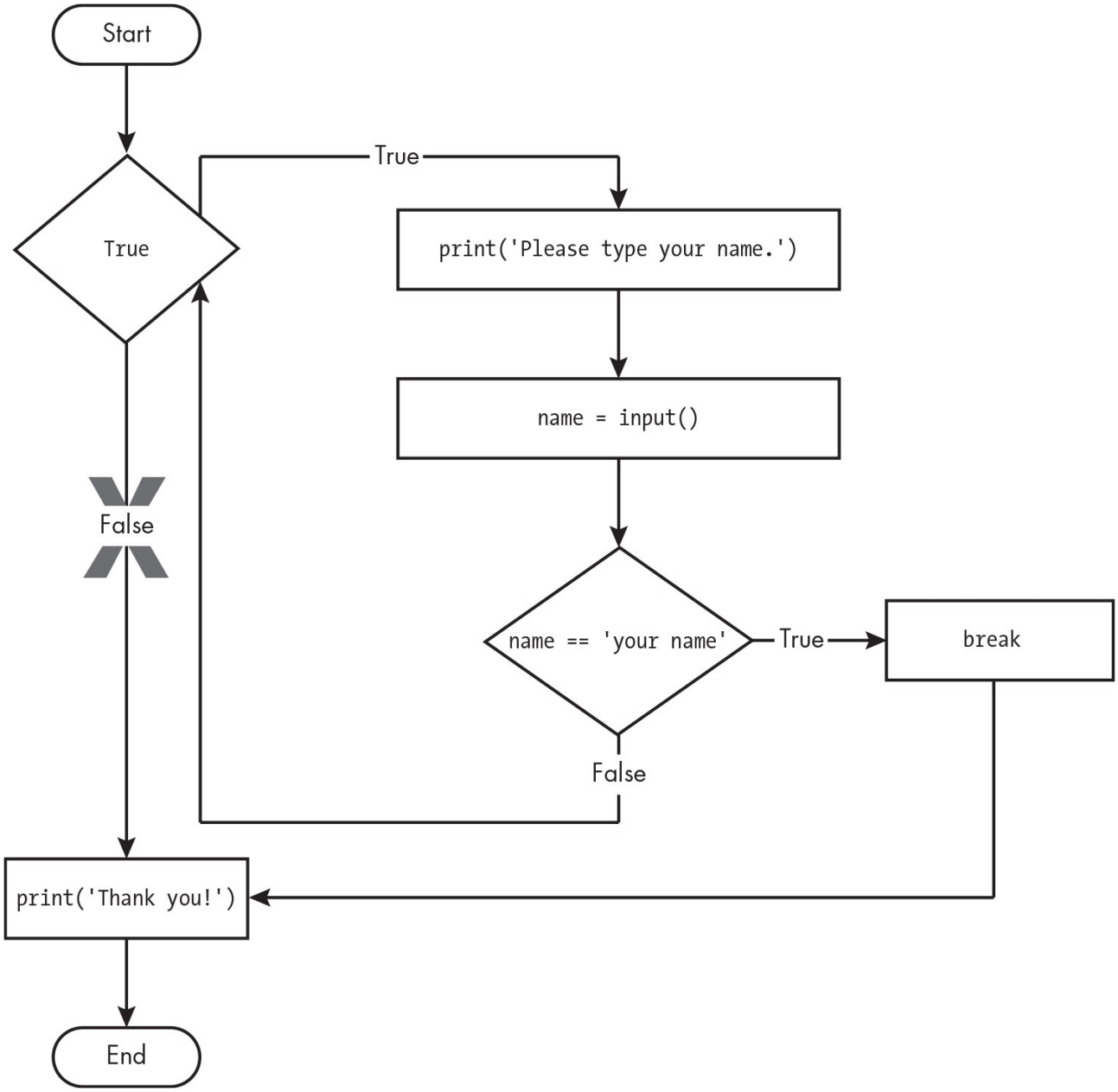
1. Saludar
2. Mostrar fecha
3. Salir

Declaraciones de break

Hay un atajo para que la ejecución del programa salga de la cláusula de un bucle de tiempo temprano. Si la ejecución llega a una instrucción de interrupción, inmediatamente sale de la cláusula del bucle while. En el código, una declaración de interrupción simplemente contiene la palabra clave break.

```
while True:
    print('Please type your name.')
    name = input('>')
    if name == 'your name':
        break
print('Thank you!')
```

La primera línea crea un *bucle* infinito; es un bucle de tiempo cuya condición es siempre verdadera. (La expresión True, después de todo, siempre evalúa el valor True.) Después de que la ejecución del programa entre en este bucle, saldrá del bucle solo cuando se ejecute una instrucción de interrupción. (Un bucle infinito que *nunca* sale es un error de programación común.)



¿ATRAPADO EN UN BUCLE INFINITO?

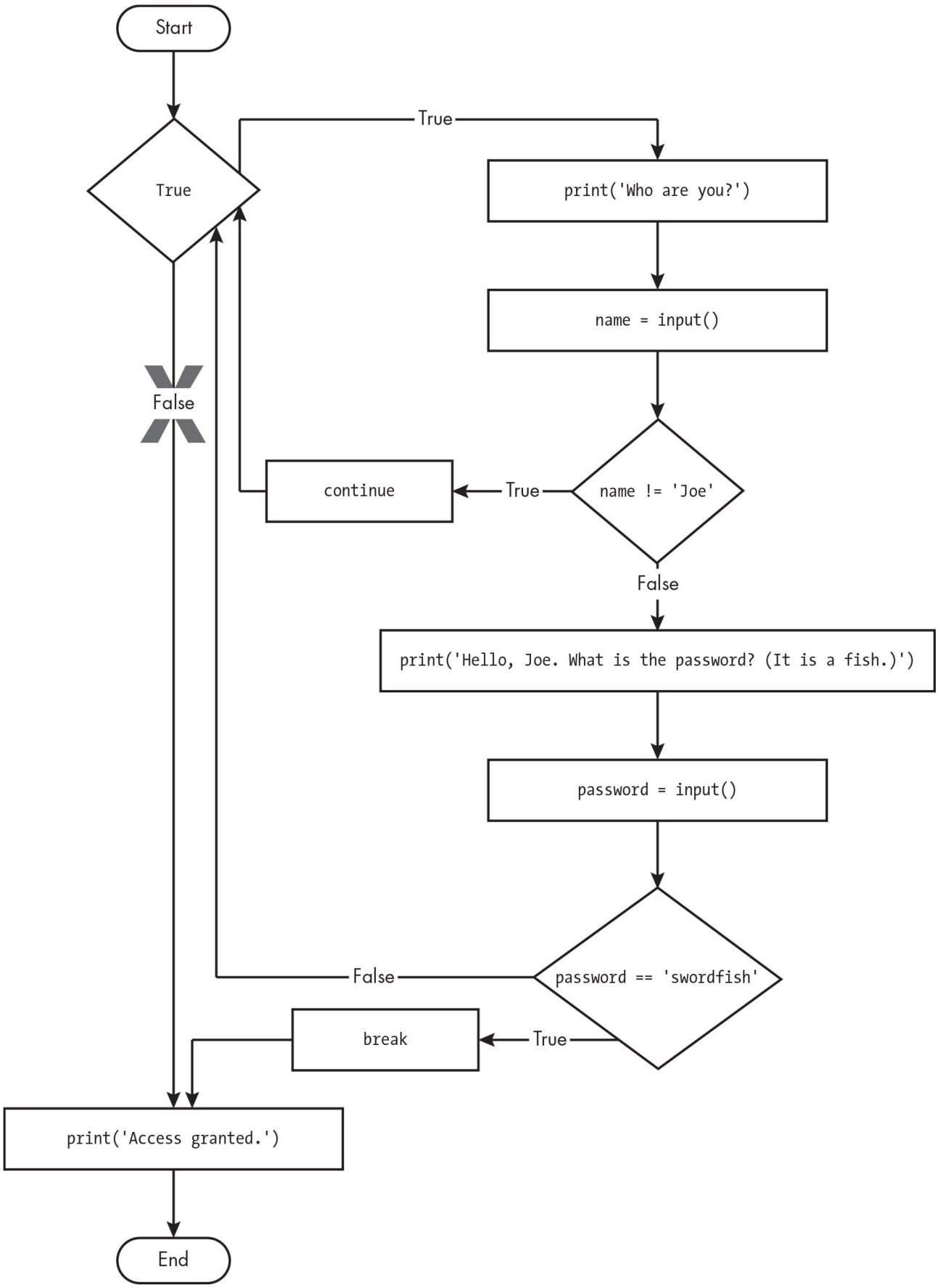
Si alguna vez ejecuta un programa que tiene un error que hace que se atasque en un bucle infinito, presione CTRL-C. Esto enviará un error de interrupción de teclado a su programa y hará que se detenga inmediatamente.

```
while True:
    print('Hello, world!')
```

Cuando ejecutes este programa, imprimirá Hello, world! A la pantalla para siempre porque la condición de la declaración de mientras siempre es verdadera. CTRL-C también es útil si simplemente desea terminar su programa de inmediato, incluso si no está atrapado en un bucle infinito.

```
while True:
    print('Who are you?')
    name = input('>')
  ❶ if name != 'Joe':
      ❷ continue
    print('Hello, Joe. What is the password? (It is a fish.)')
  ❸ password = input('>')
    if password == 'swordfish':
        ❹ break
❺ print('Access granted.')
```

Si el usuario introduce cualquier nombre además de Joe ❶, la sentencia continue ❷ hace que la ejecución del programa vuelva al inicio del bucle. Cuando el programa reevalúa la condición, la ejecución siempre entrará en el bucle, porque la condición es simplemente el valor True. Una vez que el usuario hace que pase que si se le solicita una declaración, se le pide una contraseña ❸. Si la contraseña ingresada es de pez espada, se ejecuta la instrucción break ❹ y la ejecución salta del bucle while para imprimir Access otorgado. ❺ De lo contrario, la ejecución continúa hasta el final del bucle while, donde luego salta de nuevo al inicio del bucle. Vea la Figura 3-5 para el diagrama de flujo de este programa.


Start
True
True
print('Who are you?')
name = input()
False
continue
True
name != 'Joe'
False
print('Hello, Joe. What is the password? (It is a fish.)')
password = input()
False
password == 'swordfish'
break
True
print('Access granted.')
End