ALGORITMIA

Juan Carlos Molina Lozano

Docente

OBJETIVOS DE LA CLASE

1. Comprender el uso de ciclos en programación, identificando la condición como clave de salida, y aplicar la estructura repetitiva while en la solución de problemas computacionales.
2. Aprender la sintaxis básica para en Python.
3. Resolver ejercicios para reforzar la comprensión y aplicación de la estructura repetitiva while.

METODOLOGÍA

1. Se aplicará un enfoque constructivista donde los estudiantes aprenderán a través de la exploración y la resolución de problemas.
2. Se fomentará el aprendizaje activo mediante el desarrollo de ejercicios prácticos y proyectos sencillos.
3. Se utilizarán ejemplos del mundo real para contextualizar el uso de Python.
4. Se promoverá el trabajo colaborativo a través de discusiones y resolución de problemas en grupo.
5. Se incentivará la autoevaluación y el aprendizaje basado en errores para fortalecer la comprensión del lenguaje.

CONTENIDO

1. Conceptualización del ciclo
   - ¿Qué es un ciclo en programación?
   - La condición como clave de salida
2. Estructura repetitiva while
   - Sintaxis y funcionamiento
   - Características y precauciones
3. Solución de problemas usando while
   - Ejemplos prácticos
   - Desarrollo de ejercicios
4. Resumen participativo

INTRODUCCIÓN A ESTRUCTURA REPETITIVA WHILE

Los programas informáticos frecuentemente necesitan repetir un conjunto de instrucciones múltiples veces. A este comportamiento se le conoce como ciclo o bucle. Un ciclo es una estructura de control que permite ejecutar un bloque de código repetidamente mientras se cumpla una condición lógica. En este contexto, la condición actúa como clave de salida del ciclo:

- Mientras la condición sea verdadera, el ciclo continúa.
- Cuando la condición deja de cumplirse, el ciclo se detiene.

Esta capacidad de repetir instrucciones bajo ciertas condiciones permite automatizar tareas, realizar validaciones continuas y procesar grandes volúmenes de datos de forma eficiente.

TEORÍA “MIENTRAS” EN PYTHON

- Ciclo: Conjunto de instrucciones que se repiten hasta que se cumple una condición de salida.
- Condición como clave de salida: Un ciclo necesita una condición que, al cumplirse, finaliza la repetición. Si no se cumple, el ciclo puede repetirse indefinidamente (bucle infinito).

Ejemplo real: Lavar ropa.

Mientras haya ropa sucia, seguir lavando.

Cuando no haya más ropa sucia → salir del ciclo.

Ejemplo en código (Python):

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

- ¿Qué hace que el ciclo se detenga?
- ¿Qué pasaría si no aumentamos el contador?

LA ESTRUCTURA REPETITIVA MIENTRAS (WHILE)

```
while condición:
    # bloque de instrucciones
```

Características del ciclo while

- Evalúa una condición lógica antes de ejecutar.
- Si es verdadera, ejecuta el bloque.
- Si es falsa, finaliza.
- Puede generar ciclos infinitos si no se cambia la condición adecuadamente.

Ejemplo:

```
numero = 0
while numero < 3:
    print("Hola")
    numero += 1
```

Pregunta: ¿Cómo evitar que el ciclo se repita para siempre?

VENTAJAS Y DESVENTAJAS

- ✓ Permite ejecutar instrucciones de manera repetitiva mientras se cumpla una condición.
- ✓ Es ideal cuando no se conoce de antemano cuántas veces debe repetirse el ciclo.
- ✓ Su sintaxis es simple y clara, lo que facilita su uso en programas básicos.
- ✓ Brinda flexibilidad al controlar la repetición con cualquier expresión lógica.
- x Si no se actualiza correctamente la condición, puede producir bucles infinitos.
- x Puede ser menos eficiente en situaciones donde el número de repeticiones es conocido (en ese caso, es mejor for).
- x Requiere una gestión manual de variables (contador, condición), lo que puede ser propenso a errores.
- x Es necesario tener cuidado con las condiciones de entrada, para evitar que el ciclo no se ejecute nunca.

SOLUCIÓN DE PROBLEMAS UTILIZANDO WHILE

Problema 1: Adivina el número

```
secreto = 7
numero = int(input("Adivina el número: "))
while numero != secreto:
    print("Incorrecto, intenta otra vez.")
    numero = int(input("Adivina el número: "))
print("¡Correcto!")
```

Problema 2: Menú interactivo

```
opcion = ""
while opcion != "salir":
    opcion = input("Escribe una opción ('salir' para terminar): ")
    print("Elegiste:", opcion)
```

SOLUCIÓN DE PROBLEMAS UTILIZANDO WHILE

```
numero = int(input("Ingrese un número (0 para terminar): "))
suma = 0
while numero != 0:
    suma += numero
    numero = int(input("Ingrese otro número: "))
print("La suma total es:", suma)
```

EJERCICIOS

1. Contar del 1 al 5.
2. Imprimir números pares hasta el 10.
3. Contar hacia atrás del 10 al 1.
4. Sumar números del 1 al 5.
5. Pedir un número hasta que sea mayor que 10.
6. Sumar hasta que el usuario ingrese 0.
7. Contar cuántos números ingresa el usuario hasta que escriba 0.
8. Mostrar solo números impares del 1 al 10.
9. Tabla de multiplicar del 3.
10. Adivinar un número secreto.
11. Contar cuántos números positivos y negativos ingresa el usuario (hasta 0).
12. Programa para Validar hasta que se digite una contraseña valida.
13. Sumar solo los pares hasta que se ingrese un número impar

PREGUNTAS