Clase 7: El ciclo while en un contexto visual (Turtle).

Objetivo general

- Aplicar el ciclo while para generar repeticiones controladas por condiciones, representadas gráficamente mediante un espiral en Turtle.

Objetivos particulares

1. Diferenciar el ciclo while del ciclo for.
2. Comprender la importancia de la condición de salida y la actualización de variables.
3. Usar variables acumulativas para modificar el dibujo progresivamente.
4. Crear patrones visuales dinámicos con Turtle.

Contenidos

- **Conceptuales**
    - Sintaxis y estructura de while.
    - Condición booleana y actualización de variables.
    - Prevención de bucles infinitos.
- **Procedimentales**
    - Escribir ciclos while para controlar animaciones.
    - Manipular variables para cambiar tamaño y ángulo.
    - Generar espirales con Turtle.
- **Actitudinales**
    - Desarrollar paciencia y observación en patrones progresivos.
    - Fomentar experimentación con parámetros.

¿Qué es un ciclo while?

Un **ciclo** while ejecuta un bloque de instrucciones **mientras** se cumpla una condición.

- Si la condición es verdadera → el bloque se repite.
- Si la condición se vuelve falsa → el ciclo se detiene.

💡 Piensen en un **semáforo peatonal**:

Mientras la luz esté verde → seguimos caminando.

Cuando cambia a roja → nos detenemos.

En programación, el while funciona igual: **la condición manda**.

- **Mini ejemplo en consola:**

```
contador = 0
while contador < 5:
    print("Vuelta", contador)
    contador += 1
```

- **Otro ejemplo de uso del bucle while:**

```
edad = 0
while edad < 18:
    edad = edad + 1
    print ("Felicidades, tienes: ",edad)
```

La variable edad comienza valiendo 0. Como la condición de que edad es menor que 18 es cierta (0 es menor que 18), se entra en el bucle. Se aumenta edad en 1 y se imprime el mensaje informando de que el usuario ha cumplido un año.

- **Otro ejemplo:**

Sin embargo hay situaciones en las que un bucle infinito es útil. Por ejemplo, veamos un pequeño programa que repite todo lo que el usuario diga hasta que escriba adios.

```
while True:
    entrada = input("> ")
    if entrada == "adios":
        break
    else:
        print (entrada)
```

Comprobamos entonces si lo que escribió el usuario fue adios, en cuyo caso se ejecuta la orden break o si era cualquier otra cosa, en cuyo caso se imprime en pantalla lo que el usuario escribió. **La palabra clave break (romper) sale del bucle en el que estamos.**

Diferencia con for

Característica	for	while
Número de repeticiones	Lo sabemos desde el inicio.	No lo sabemos, depende de la condición.
Uso típico	Recorrer listas, repetir un número fijo de veces.	Repetir hasta que algo cambie.
Ejemplo cotidiano	"Hacer 10 flexiones" → sabemos cuántas.	"Hacer flexiones hasta cansarse" → no sabemos cuántas serán.

Ejemplo guiado – Espiral básico (while + turtle)

```
import turtle as t

t.speed(0)  # Velocidad máxima
paso = 10

while paso < 200:
    t.forward(paso)
    t.right(45)
    paso += 5

t.done()
```

Ejercicio guiado – Espiral multicolor

```
import turtle as t

t.speed(0)
paso = 5
colores = ["red", "blue", "green", "purple", "orange", "pink"]
i = 0

while paso < 300:
    t.color(colores[i % len(colores)])
    t.forward(paso)
    t.right(30)
    paso += 3
    i += 1

t.done()
```

- **Uso del operador % para repetir colores de forma cíclica.**

- **t.color(colores[i % len(colores)])**

  Este es un truco matemático muy común:

- `len(colores)` → devuelve cuántos colores hay en la lista (6 en este caso).
- `i % len(colores)` → calcula el **resto** de dividir `i` por 6.
  - Cuando `i = 0` → `0 % 6 = 0` → usa `"red"`
  - `i = 1` → `1 % 6 = 1` → `"blue"`
  - …
  - `i = 6` → `6 % 6 = 0` → vuelve a `"red"`

Así, el operador % (**módulo**) hace que los colores se repitan en un **bucle infinito**.

Código Python: Espiral limitada con while

```
import turtle  # Importamos la librería Turtle

#Configuración inicial de la ventana
turtle.speed(0)       # Velocidad máxima del dibujo
turtle.bgcolor("black")  # Fondo negro
turtle.pencolor("cyan")  # Color del lápiz

#Variables de control
longitud = 2    # Longitud inicial de cada línea
giro = 20       # Ángulo de giro en grados
limite = 200    # Longitud máxima antes de detener el bucle

#Ciclo while: dibujar mientras la longitud sea menor que el límite
while longitud < limite:
    turtle.forward(longitud)  # Avanza la longitud actual
    turtle.right(giro)        # Gira a la derecha el ángulo indicado
    longitud += 2             # Aumenta la longitud para que la espiral crezca

#Finaliza el programa al hacer clic en la ventana
turtle.done()
```

Explicación línea por línea

```
import turtle
```

Importa la librería **Turtle** que nos permite dibujar de forma visual en una ventana.

Configuración inicial:

- turtle.speed(0) → Ajusta la velocidad de dibujo al máximo (0 es "sin pausa").

- turtle.bgcolor("black") → Cambia el fondo de la ventana a negro.
- turtle.pencolor("cyan") → Establece el color del trazo en cian.

Variables de control:

- longitud = 2 → Define el tamaño inicial de cada segmento de línea.
- giro = 20 → Ángulo fijo en grados que el "lápiz" girará en cada paso.
- limite = 200 → Condición que usaremos para detener el bucle (while dejará de repetirse cuando longitud llegue a 200).

El ciclo while:

while longitud < limite:

- Traducción: **"Mientras la longitud sea menor que el límite, sigue repitiendo las instrucciones"**.
- Esto lo diferencia de un for, que tiene un número fijo de repeticiones. Aquí, el ciclo depende de una **condición que puede cambiar**.

Dentro del ciclo:

1. turtle.forward(longitud) → Mueve la tortuga hacia adelante el número de píxeles indicado por longitud.
2. turtle.right(giro) → Gira el ángulo definido, en este caso 20°, lo que va formando la espiral.
3. longitud += 2 → Aumenta la longitud para que cada segmento sea un poco más largo que el anterior, creando el efecto de espiral.

Finalización:

turtle.done()

- Mantiene la ventana abierta hasta que el usuario la cierre manualmente (o haga clic en ella).
- Sin esto, la ventana se cerraría inmediatamente después de terminar el ciclo.

Código Python: Espiral limitada con for

```
import turtle  # Importamos la librería Turtle

# Configuración inicial
turtle.speed(0)        # Velocidad máxima
turtle.bgcolor("black") # Fondo negro
turtle.pencolor("cyan") # Color del lápiz

# Variables de control
giro = 20       # Ángulo de giro en grados
limite = 200    # Longitud máxima de la espiral
paso = 2        # Cuánto crece la longitud en cada paso

# Ciclo for: sabemos cuántas repeticiones habrá
for longitud in range(2, limite, paso):
    turtle.forward(longitud)  # Avanza la longitud actual
    turtle.right(giro)        # Gira el ángulo indicado

# Mantener la ventana abierta
turtle.done()
```

En la versión con while:

- Teníamos una variable longitud que empezaba en 2 y la íbamos aumentando con longitud += 2.
- El ciclo dependía de la condición longitud < limite.
- **En la versión con for**:
- Usamos range(2, limite, paso), que **genera todos los valores de longitud** desde 2 hasta el límite, avanzando de paso en paso.
- El ciclo sabe desde el principio cuántas repeticiones hará.