Los Ciclos

El ciclo while

El ciclo while es muy sencillo, solo ponemos la condición a evaluar y en su interior el código que se ejecuta mientras la condición se cumpla; manteniendo las reglas de sangría ya conocidas

Sintaxis

Haz doble clic (o pulsa Intro) para editar

```
while condicion:
    codigo
    codigo
```

Ejemplo Ciclo while

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

```
color = "rojo"
while color != "blanco":
  color = input("Ingrese color")
  print ("El color seleccionado fue", color)
```

El ciclo for

La sintaxis general del ciclo for es

for nombre_variable **in** iterable**:**

```
for variable in iterable:
    codigo
    codigo
```

```
cad = [1,23,5,22,80, "casa", 2.4, "sena", 21]
for x in cad:
  print (x)
```

```
1
23
5
22
80
casa
2.4
sena
21
```

```
for x in range(5): #0,1,2,3,4
  print ("Hola ",x+1)
```

Es común usar la función **range()** para generar el iterable deseado, en caso de querer ejecutar el ciclo for un número determinado de veces.

La función **range()** tiene los siguientes parámetros: range(start, stop, step).

start: (opcional) valor inicial, por defecto 0

stop: (obligatorio) posición final, detiene el ciclo antes de llegar a ella

step: (opcional) incremento en cada paso, por defecto 1

Ejemplo

Haz doble clic (o pulsa Intro) para editar

```
for x in range (1, 11, 2):
    print (x)
```

```
for x in range (0,5,1):
  print ("hola", x)
```

```
for x in range (2, 12, 2):
    print (x)
```

```
for x in range (10):
    print (x)
```

```
for x in range (9,0,-1):
    print (x)
```

```
for x in range (3,100,3):
    print (x)
```

```
nombre = input("ingrese nombre:")
for x in range (1, len(nombre), 2): # 1,3,5,7
  print (x)
```

break y continue

Podemos salir abrúptamente de un ciclo (break) o pasar abrúptamente a la ejecución del siguiente paso del ciclo a verificar el cumplimiento de la condición (continue)

¿Qué mostraría el siguiente programa?

```
print("La instrucción break:")
for i in range(1, 6): #[1,2,3,4,5]
    if i == 3:
        break

    print("Dentro del bucle.", i)

print("Fuera del bucle.")
```

```
La instrucción break:
Dentro del bucle. 1
Dentro del bucle. 2
Fuera del bucle.
```

¿Qué mostraría el siguiente programa?

```
# continue - ejemplo

print("\nLa instrucción continue:")
for i in range(1, 6):
```

```
    if i == 3:
        continue
    print("Dentro del bucle.", i)
print("Fuera del bucle.")
```

Apropiación [Ejercicios Cíclicos]

1. Elabore un algoritmo para que los niños practiquen las tablas de multiplicar. El programa debe solicitar al niño que tabla quiere repasar y luego hacerle 6 preguntas aleatorias de esa tabla. Si el niño responde correctamente, se le felicitará y animará. Si responde incorrectamente, se le indicará su error. Al finalizar las 6 preguntas, se le dirá cuántas respondió correctamente. Si acierta al menos 3 preguntas, se le sugiere que siga practicando esa tabla y se le muestra la tabla completa. Si responde correctamente a todas las preguntas, se le indicará que se sabe la tabla y se le invitara a que siga repasando otras tablas de multiplicar. Si responde incorrectamente a menos de 3 preguntas, se le mostrará la tabla completa y se le preguntará si quiere volver a estudiarla o salir.

```
import random

tabla = int(input("¿Qué tabla quieres repasar? (del 1 al 10): "))
correctas = 0
contador = 0

while contador < 6:
    numero = random.randint(1, 10)
    respuesta = int(input(f"¿Cuánto es {tabla} x {numero}? "))
    if respuesta == tabla * numero:
        print("¡Correcto! Muy bien, sigue así 😊")
        correctas += 1
    else:
        print(f"Incorrecto. La respuesta correcta es {tabla * numero}.")
    contador += 1

print(f"\nRespondiste correctamente {correctas} de 6 preguntas.")

if correctas == 6:
    print("¡Excelente! Te sabes la tabla muy bien. Ahora puedes repasar otras tablas.")
elif correctas >= 3:
    print("Buen trabajo, sigue practicando esta tabla para mejorar aún más.")
    print(f"Tabla del {tabla}:")
    i = 1
    while i <= 10:
        print(f"{tabla} x {i} = {tabla * i}")
        i += 1
else:
    print("Es recomendable que sigas estudiando esta tabla.")
    print(f"Tabla del {tabla}:")
    i = 1
    while i <= 10:
        print(f"{tabla} x {i} = {tabla * i}")
        i += 1
    opcion = input("¿Quieres volver a estudiarla? (sí/no): ").lower()
    if opcion == "sí" or opcion == "si":
        # Aquí podrías repetir el programa pero sin funciones es necesario ejecutar todo de nuevo o hacer un loop externo.
        print("Repite el programa para volver a estudiar la tabla.")
    else:
        print("¡Hasta luego! Sigue practicando.")
```

2. Se requiere desarrollar un algoritmo que permita que los asistente a un partido de fútbol puedan saber cuanto pagaran por su entrada, se ofrece un descuento a los aficionados que depende del estrato y la edad. Si el estrato es 1 y su edad es menor a 18 el descuento será del 20% sobre el valor de la boleta. Si el estrato es 1 y el aficcionado tiene 18 o mas años, el descuento será del 15%. Si el estrato es 2 y la edad es menor a 18 años, el descuento será del 10% y si el estrato es 2 y la edad es 18 años o más, el descuento será del 5%. Determinar el total del dinero recaudado, el total de personas por estrato que ingresaron al partido, el total de mayores y menores de edad de acuerdo a las N personas que ingresan al partido.

3. Leer un password de ingreso a un programa y mostrar el mensaje de bienvenida si es correcto. Mientras no lo sea, debe mostrar el mensaje de Password incorrecto. El programa debe terminar automáticamente al quinto intento fallido.

4. Pide al usuario que ingrese un número, y luego indiquele cuántos dígitos tiene en número.

5. La secuencia Fibonacci es una serie de números en la que cada término es la suma de los dos términos anteriores. La secuencia comienza con 0 y 1, y luego cada término sucesivo se calcula sumando los dos términos anteriores. Por ejemplo, el tercer término es la suma de los dos términos anteriores (0 + 1), por lo que es 1. El cuarto término es la suma de los dos términos anteriores (1 + 1), por lo que es 2. Y así sucesivamente.

Elabore un algoritmo que Genere los primeros 20 términos de la secuencia Fibonacci (0, 1, 1, 2, 3, 5, 8, 13, 21, ...).

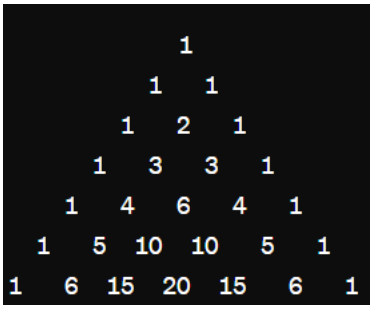
6. Solicita al usuario que ingrese una secuencia de números separados por comas. Cuenta cuántos números pares e impares hay en la secuencia.

7. Pide al usuario que ingrese dos números y verifica si son amigos o no. Dos números son amigos si la suma de los divisores propios de uno es igual al otro número y viceversa.

8. Solicita al usuario que ingrese un número y calcula la suma de los factoriales de los números del 1 al número ingresado.

9. Elabore un algoritmo utilizando el ciclo While que permita simular un temporizador simple que cuenta hacia atras desde un número ingresado por el usuario hasta 0.

10. El Triángulo de Pascal es una estructura triangular de números que se utiliza en combinatoria y álgebra, donde cada número en el triángulo es la suma de los dos números directamente encima de él en la fila anterior.



Elabore un algoritmo que genera las primeras 10 filas del Triángulo de Pascal.