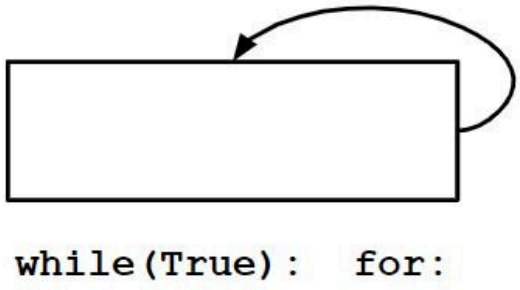
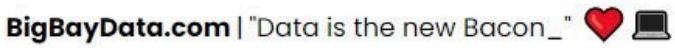
Estructuras Repetitivas





> Link: https://www.bigbaydata.com/ejercicios-de-bucles-en-python/

Objetivos

- Entender el uso de range()
- Principios de la estructura del for
- El for iterable
- El uso de While
- While vs For: Breve análisis y cuando usar cada una
- Errores comunes

Bucle For

> Tanto el `for` como el `while` son estructuras, por eso, deberíamos ir familiarizándonos a los `:`.

En el caso del for, comunmente buscaremos definir un inicio y un criterio de parada. En general, podemos utilizar la función range() que acompaña a for. Fíjate, para aprender como funciona `range()` podemos jugar con print y hacer unas pruebas:

In [ ]:
```
print(range(5)) # Si no colocamos nada, empieza en 0.
```

In [ ]:
```
# For tiene la particularidad de ir desde 0 hasta N - 1, siendo N el número que

for i in range(5):
    print("Iteración", i)
```

```
# El código print() es ejecutado varias veces, donde i pasa a valer, en distinta
# Una vez llega al valor máximo, sigue el código.
```

In [ ]:
```
suma = 0
for i in range(1, 101):
    #print(i)
    suma += i # A esto lo llamamos sobrecarga. Es exactamente igual que hacer su
    #print(suma, i)
print("La suma total es:", suma)

# Descomenta los print() para entender lo que está pasando... Juega a bajar el n
```

In [ ]:
```
# Otro uso de range()...
for i in range(2, 20, 2): # Podemos ir saltando de 2 en 2 también...
    print(i)
```

For iterable

> Otro uso común hilado con los temas posteriores (Listas y Textos) es utilizar el for con **elementos iterables / colecciones**. Por ahora, ejecútalo y observa cómo se puede utilizar:

In [ ]:
```
# Segundo uso típico de For: X pasa a ser cada elemento de dentro de 'cursos'.
# Igual que antes, comienza en el primer elemento y termina en el último:

cursos = ["IA", "Big Data", "SQL", "Python", "SEO"]
for x in cursos:
  print(x)
```

Bucle While - Con ejemplo

While es una estructura repetitiva que se usa para la misma idea que el for: repetir instrucciones.

Observa su funcionamiento, después reflexionaremos por qué hace falta:

In [ ]:
```
contador = 1
# Los códigos se ejecutarán si es True la condición. Juega con contador a coloca
while contador <= 5:
    print("Contador:", contador)
    contador += 1
```

In [ ]:
```
suma = 0
numero = 2
while numero <= 10:
    suma += numero
    numero += 2

print("La suma de números pares hasta 10 es:", suma)
# Como ves, igual que con for, el segundo nivel puede tener tantas líneas como q
```

¿Cómo funciona While?

Si estás comenzando, **cuidado con while**. Al principio, cuesta entender el funcionamiento. Para practicarlo, las claves son principalmente 2:

- 1. El `while` funciona con condiciones ciertas o falsas. Se elaboran parecido a las condicionales.
- 2. Debemos tener cuidado con los posibles `bucles infinitos`.

Haz lo siguiente:

Ve a https://www.online-python.com/ y pega este código. Si te gusta, puedes practicar también ahí.

En esa web programaba con los alumnos las primeras veces, es fiable. Ejecútalo y verás lo que sucede al de un rato. La web caerá...:

In [ ]:
```
a = 1
while(True):
    a += 1 # Es lo mismo que a = a + 1...
    print(a)
```

Moraleja: Cuidado con los bucles while. Asegúrate de que la condición no se quede siempre a cierto, sino, tendrás un efecto parecido al de arriba :)

While vs For: Breve análisis

- Imaginemos un problema donde tenemos que evaluar 11 jugadores de una Base de Datos. En ese caso, el número es conocido. Se usa `for`.
- Imaginemos que queremos hacer una calculadora para que sea usado **hasta** que el usuario teclee 'S'. Aquí, tenemos a `while`.
- Imaginemos que buscamos diseñar el acceso con pin, donde el usuario tiene **hasta** 3 intentos. `While` nuevamente.

> El While es diferente al for. En la práctica, es clave saber identificar si deberíamos usar `while` o `for`.

> Como conclusión, si tenemos claro cuántas veces queremos repetir un código está for. Sino, si identificamos alguna condición o intuimos la palabra *hasta* tendremos a While.

Errores más comunes

In [ ]:
```
# 1. Bucle infinito.

# Ejemplo de bucle infinito
i = 0
while i < 5:
    print(i)
    # Olvidamos incrementar 'i'
# Solución: Asegúrate de actualizar la variable de control. Con for no tendrás e
```

```
i = 0
while i < 5:
    print(i)
    i += 1  # Incrementamos 'i'
```

In [ ]:
```
# 2. Condición nunca verdadera (no se ejecuta while nunca)

# Ejemplo de condición nunca verdadera
i = 10
while i < 5:
    print(i)
    i += 1
# Solución: Verifica las condiciones iniciales.
i = 0
while i < 5:
    print(i)
    i += 1
```

In [ ]:
```
#3. Usar el for de iteración y eliminar elementos de la lista

# Ejemplo de modificar la lista durante la iteración
numbers = [1, 2, 3, 4, 5]
for num in numbers:
    if num == 3:
        numbers.remove(num)
print(numbers)  # Salida inesperada: [1, 2, 4, 5]
# Solución: Iterar sobre una copia de la lista.
numbers = [1, 2, 3, 4, 5]
for num in numbers[:]:  # Usar una copia de la lista
    if num == 3:
        numbers.remove(num)
print(numbers)  # Salida correcta: [1, 2, 4, 5]
```

Ejercicios resueltos

In [ ]:
```
#Ej1.

for i in range(1, 10 + 1):
    print(i)
```

In [ ]:
```
#Ej2.

# 5! = 5 * 4 * 3 * 2 * 1
n = int(input('Dame el valor del factorial a calcular...'))
base = 1 #Colocamos a 1 porque algo * 1 es ese mismo número.
for i in range(1, n + 1):
    base *= i
print(base)
```

In [ ]:
```
#Ej3.

frase = input('Dame una frase...')
frase = frase.split(' ')
for p in frase:
    print(p)
print( len(frase) )
```

In [ ]:
```python
#Ej4.

num = 1
num1 = 0
num2 = 1
n = int(input('Dime el valor de N...'))

for i in range(0, n):
    print(num)
    num = num1 + num2
    num1 = num2
    num2 = num
```

In [ ]:
```python
#Ej5.

num = int(input('Dime el número de * que quieres...'))

for i in range(0, num + 1):
    print(i * '*')
```

In [ ]:
```python
#Ej6.

op = int(input('Dame un número... - {-1 para finalizar}'))
sum = 0
while(op != -1): #Fíjate en el juego: 1. Buscamos colocar la condición de repeti
    sum += op
    print(sum)
    op = int(input('Dame un número... - {-1 para finalizar}')) #Al final, volvem
```

In [ ]:
```python
#Ej7.

PIN_SECRETO = '1234' #Este tipo de variables se consideran globales. Como no sue
# Otros ejemplos: URL_BASEDATOS, DNI_CLIENTE...

pin = input('Dame el PIN...')
intentos = 0
#Los 3 intentos máximos...
while(pin != PIN_SECRETO and intentos < 2): # El bucle básicamente fuerza a rein
    intentos += 1
    print('Pin incorrecto. Intentos:',intentos,', Vuelve a intentarlo...')
    pin = input('Dame el PIN...') #Punto4: Aquí termina el bucle. Si la condició

# Según qué ha pasado...
if(pin==PIN_SECRETO):
    print('Desbloquear Pantalla')
else:
    print('Llamando a @policia...')
```

In [ ]:
```python
#Ej8.

import math

# Pedir el primer cateto
cateto1 = float(input("Introduce el valor del primer cateto (debe ser mayor a 0)
while cateto1 <= 0:
    print("El valor debe ser mayor a 0. Inténtalo de nuevo.")
    cateto1 = float(input("Introduce el valor del primer cateto (debe ser mayor
```

```
# Pedir el segundo cateto
cateto2 = float(input("Introduce el valor del segundo cateto (debe ser mayor a 0
while cateto2 <= 0:
    print("El valor debe ser mayor a 0. Inténtalo de nuevo.")
    cateto2 = float(input("Introduce el valor del segundo cateto (debe ser mayor

# Calcular la hipotenusa
hipotenusa = math.sqrt(cateto1**2 + cateto2**2)

# Mostrar el resultado
print("La hipotenusa es:",hipotenusa)
```

In [ ]:
```
#Ej9.

import math

operacion = input('Operación? Teclea 1, 2, 3 ó SAL') #recogemos el valor de la o
resultado = 0 # Dejamos la variable resultado inicializada para ser utilizada hi
while(not operacion=='SAL'):
    inputA = float(input('Dame el valor de A'))
    inputB = float(input('Dame el valor de B'))
    if(operacion=='1'):
        resultado = math.sqrt(inputA+inputB)
        print(resultado)
    elif(operacion=='2'):
        if(not inputB == 0): #Solo si no divide entre 0...
            resultado = inputA/inputB
            print(resultado)
        else:
            print('Error')
    else: #El último caso es la operación 3...
        resultado = (inputA+inputB)/2.5
        print(resultado)
    operacion = input('Operación? Teclea 1, 2, 3 ó SAL') #recogemos el valor de
print('Saliendo del programa...')
```

In [ ]:
```
#Ej10.

#Estadisticas de cada personaje: Ataque y Defensa
# El código más adelante verás que se puede mejorar. La idea es seguir asentando
import random, time
# Se usan dos librerías como novedad: una es para generar números aleatorios. La

vidaA, ataqueA = 100, 20
vidaB, ataqueB = 100, 20

turno = random.randint(-1,1) #Se utilizan -1 o 1 para establecer quien empieza.

while(vidaA > 0 and vidaB > 0): #Bucle de combate por turnos...
    if(turno == 1): #Si 1 entonces ataque de B que resta a la vida de A.
        vidaA = vidaA - ataqueB
    else:
        vidaB = vidaB - ataqueA
    turno = turno * (-1) # 1 * (-1) = -1. Cambio de turno en el bucle. #Siguient
    print('Vida de A:',vidaA, 'Vida de B:',vidaB)
    time.sleep(2) #El sistema se detiene 2 segundos para ver el progreso del com
    # Y así hasta que no sea cierto que ambos estén vivos. Todo lo anterior está
```

```
if(vidaA > 0):
    print('Ha ganado A')
else:
    print('Ha ganado B')
```