

ESTRUCTURAS REPETITIVAS

En Python, las estructuras repetitivas (también llamadas bucles o loops) son instrucciones que permiten ejecutar un bloque de código varias veces, dependiendo de una condición o una secuencia de elementos. Los tipos de estructura repetitivo en Python son:

1. While (mientras).

Ejecuta un bloque de código mientras una condición sea verdadera.

Ejemplo: contador=0

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

2. For (por).

Se utiliza para iterar una secuencia como una lista, tupla, string o rango de números.

Ejemplo: `for i in rang (5)`
`print(i)`

ejemplo N°1:

clase: Factorial

atributos: Numero, resultado

acción: calcular

objeto: miFactorial =Factorial(10)

```
class Factorial:
    def __init__(self, numero):
        self.numero = numero
        self.resultado = 1

    def calcular(self):
        if self.numero < 0:
            print("El factorial no esta definido para numeros negativos")
            return None
        for i in range(1,self.numero + 1):
            self.resultado *= i
        return self.resultado

def main():
    miFactorial = Factorial(5)
    resultado = miFactorial.calcular()
    if resultado is not None:
        print(f"El factorial de {miFactorial.numero}es{resultado}")

if __name__=="__main__":
    main()
```

ejemplo N°2:

clase: fibonacci

atributos: cantidad, serie

acción: generar serie

objeto: mi fibonacci= Fibonacci(10)

```
class Fibonacci:
    def __init__(self, cantidad):
        self.cantidad = cantidad
        self.serie=[]

    def generarSerie(self):
        a,b = 0,1
        for _ in range(self.cantidad):
            self.serie.append(a)
            a, b =b, a + b
        return self.serie

def main():
    cantidad = int(input("Ingrese la cantidad de la serie:"))
    miFibonacci = Fibonacci(cantidad)
    resultado = miFibonacci.generarSerie()
    print(resultado)

if __name__=="__main__":
    main()
```

ejemplo N°3:

clase: multiplicacion

atributos: numero

acción:

objeto:

```
class Multiplicacion:
    def __init__(self, numero):
        self.numero = numero

    def generar_tabla(self):
        print(f"Tabla de multiplicar del {self.numero}")
        for i in range(1, 11):
            resultado = self.numero * i
            print(f"{self.numero} x {i} = {resultado}")

def main():
    numero = int(input("Ingrese un número para ver su tabla de multiplicar: "))
    tabla = Multiplicacion(numero)
    tabla.generar_tabla()

if __name__=="__main__":
    main()
```

ejemplo N°4:

clase: sumaNaturales

atributos : limites

acción:

objeto:

```
class SumaNaturales:
    def __init__(self, limite):
        self.limite = limite
        self.suma_pares = 0
        self.suma_impares = 0
        self.suma_total = 0

    def calcularSuma(self):
        print("Lista de números y si son pares o impares:")
        for i in range(1, self.limite + 1):
            self.suma_total += i
            if i % 2 == 0:
                self.suma_pares += i
                print(f"{i} es PAR")
            else:
                self.suma_impares += i
                print(f"{i} es IMPAR")
        return self.suma_total, self.suma_pares, self.suma_impares

def main():
    miSuma = SumaNaturales(10)
    resultado_total, resultado_pares, resultado_impares = miSuma.calcularSuma()
    print("-" * 30)
    print(f"La suma de los primeros {miSuma.limite} números naturales es: {resultado_total}")
    print(f"La suma de los números pares es: {resultado_pares}")
    print(f"La suma de los números impares es: {resultado_impares}")

if __name__ == "__main__":
    main()
```

ejemplo N°5:

clase: pitagoras

atributos: catettoa, catetob,

acción:

objeto:

```
class Pitagoras:
    def __init__(self, catettoa, catetob):
        self.catettoa = catettoa
        self.catetob = catetob
```

```
def calcular_hipotenusa(self):  
    return (self.catetoa**2 + self.catetob**2)**0.5  
  
catetoa = float(input("Ingrese el valor del cateto a: "))  
catetob = float(input("Ingrese el valor del cateto b: "))  
  
operacion = Pitagoras(catetoa, catetob)  
print("La hipotenusa es", operacion.calcular_hipotenusa())
```