

**Universidad Nacional del Altiplano**  
**Facultad de Ingeniería Estadística e Informática**

**Docente:**

Ing. Coyla Idme Leonel

**Alumno:**

Ticona Miramira Roberto Angel

**Estructuras repetitivas**

**» DESCRIPCIÓN**

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 estructuras repetitivas en Python son:

- While, ejecuta un bloque de código mientras una condición sea verdadera.
- For, se utiliza para iterar una secuencia como una lista, tupla, string o rango de números.

**» EJERCICIO 1**

Determinar el factorial de un número.

- Clase: Factorial.
- Atributo: Número.
- Acción: Calcular.
- Objetos: miFactorial = Factorial(5)

**Código**

```
1 class Factorial:
2     def __init__(self, numero):
3         self.numero = numero
4         self.resultado = 1
5
6     def calcular(self):
7         if self.numero < 0:
8             print("El factorail no esta definido para número negativos")
9             return None
10
11         for i in range(1, self.numero + 1):
12             self.resultado *= i
13         return self.resultado
14
15 # Crear main
16
17 def main():
18     miFactorial = Factorial(5)
19     resultado = miFactorial.calcular()
20     if resultado is not None:
21         print(f"El factorial de {miFactorial.numero} es: {resultado}")
22 if __name__ == "__main__":
23     main()
```

**Ejecución**

```
1 El factorial de 5 es: 120
```

## » EJERCICIO 2

Determinar la serie de Fibonacci

- Clase: Fibonacci.
- Atributos: cantidad, serie
- Acción: generarSerie.
- Objeto: miFibonacci(10).

### Código

```
1 class Fibonacci:
2     def __init__(self, cantidad):
3         self.cantidad = cantidad
4         self.serie = []
5
6     def generarSerie(self):
7         a, b = 0, 1
8         for _ in range(self.cantidad):
9             self.serie.append(a)
10            a, b = b, a + b
11        return self.serie
12
13 def main():
14     cantidad = int(input("Ingrese la cantidad de la serie: "))
15     miFibonacci = Fibonacci(cantidad)
16     resultado = miFibonacci.generarSerie()
17     print(resultado)
18
19 if __name__ == "__main__":
20     main()
```

### Ejecución

```
1 Ingrese la cantidad de la serie: 10
2 [0, 1, 1, 2, 3, 5, 8, 13, 21, 34]
```

## » EJERCICIO 3

Crear una tabla de multiplicar

- Clase: Tabla.
- Atributos: Numero.
- Acción: Multiplicación.
- Objetos: Resultado.

### Código

```
1 class Tabla:
2     def __init__(self, numero):
3         self.numero = numero
4
5     def multiplicacion(self):
6         for i in range(1, 11):
7             resultado = self.numero * i
```

```
8         print(f"{self.numero} X {i} = {resultado}")
9
10 def main():
11     numero = int(input("Ingrese un número: "))
12     resultado = Tabla(numero)
13     resultado.multiplicacion()
14 if __name__ == "__main__":
15     main()
```

### Ejecución

```
1 Ingrese un número: 8
2 8 X 1 = 8
3 8 X 2 = 16
4 8 X 3 = 24
5 8 X 4 = 32
6 8 X 5 = 40
7 8 X 6 = 48
8 8 X 7 = 56
9 8 X 8 = 64
10 8 X 9 = 72
11 8 X 10 = 80
```

### » EJERCICIO 4

Determinar la suma de números naturales.

- Clase: SumaNaturales.
- Atributo: limite, suma.
- Acción: calcularSuma().
- Objeto: miSuma = SumaNaturales(10).

### Código

```
1 class SumaNaturales:
2     def __init__(self, limite):
3         self.limite = limite
4         self.suma = 0
5
6     def calcularSuma(self):
7         for i in range(1, self.limite + 1):
8             self.suma = self.suma + i
9         return self.suma
10
11 def main():
12
13     miSuma = SumaNaturales(10)
14     resultado = miSuma.calcularSuma()
15     print(f"La suma de los primeros {miSuma.limite} números naturales es
16           ↳ {resultado}")
17
18 if __name__ == "__main__":
19     main()
```

### Ejecución

```
1 La suma de los primeros 10 números naturales es 55
```

### » EJERCICIO 5

Determinar en una lista de números consecutivos que números son pares o impares.

- Clase: Números.
- Atributo: Número.
- Acción: Clasificar().
- Objeto: misNumeros(10).

### Código

```
1 class Numeros:
2     def __init__(self, numero):
3         self.numero = numero
4
5     def clasificar(self):
6         for i in range (0, self.numero + 1):
7             if i == 0:
8                 print("El número es 0")
9             elif i % 2 == 0:
10                print(f"{i} es par")
11            else:
12                print(f"{i} es impar")
13
14 def main():
15
16     misNumeros = Numeros(10)
17     resultado = misNumeros.clasificar()
18
19 if __name__=="__main__":
20     main()
```

### Ejecución

```
1 El número es 0
2 1 es impar
3 2 es par
4 3 es impar
5 4 es par
6 5 es impar
7 6 es par
8 7 es impar
9 8 es par
10 9 es impar
11 10 es par
```

### » EJERCICIO 6

Crear un programa que permita ingresar una cierta cantidad de datos y determine el promedio, la varianza y la desviación estándar de todos los datos ingresados.

- Clase: Estadística.
- Atributo: Cantidad.
- Acción: promedio(), varianza1P(), varianza2P(), varianza1M(), varianza2M().
- Objeto: misCalculos = Estadistica(cant)

### Código

```
1 class Estadistica:
2     def __init__(self, cantidad):
```

```
3         self.cantidad = cantidad
4         self.datos = []
5
6     def promedio(self):
7         suma = 0
8         for i in range(1, self.cantidad + 1):
9             dato = float(input(f"Dato {i}: "))
10            self.datos.append(dato)
11            suma += dato
12
13        return suma / self.cantidad
14
15    def varianza1P(self):
16        prom = sum(self.datos) / self.cantidad
17        suma_dif = 0
18        for x in self.datos:
19            suma_dif += (x - prom) ** 2
20        return suma_dif / self.cantidad
21
22    def varianza2P(self):
23        suma = sum(self.datos)
24        suma_cuadrados = 0
25        for z in self.datos:
26            suma_cuadrados += z ** 2
27        return (suma_cuadrados - (suma ** 2) / self.cantidad) / self.cantidad
28
29    def varianza1M(self):
30        prom = sum(self.datos) / self.cantidad
31        suma_dif = 0
32        for x in self.datos:
33            suma_dif += (x - prom) ** 2
34        return suma_dif / (self.cantidad - 1)
35
36    def varianza2M(self):
37        suma = sum(self.datos)
38        suma_cuadrados = 0
39        for z in self.datos:
40            suma_cuadrados += z ** 2
41        return (suma_cuadrados - (suma ** 2) / self.cantidad) / (self.cantidad - 1)
42
43    def main():
44
45        cant = int(input("Ingrese la cantidad de datos: "))
46        misCalculos = Estadistica(cant)
47        promedio = misCalculos.promedio()
48
49        varianza1 = misCalculos.varianza1P()
50        varianza2 = misCalculos.varianza2P()
51        varianza1M = misCalculos.varianza1M()
52        varianza2M = misCalculos.varianza2M()
53
54        print(f"El Promedio es: {promedio}")
55        print(f"La Varianza Poblacional calculada con la formula 1 es: {varianza1}")
56        print("Su Desviación Estándar es: ", varianza1 ** (1/2))
57        print(f"La Varianza Poblacional calculada con la formula 2 es: {varianza2}")
58        print("Su Desviación Estándar es: ", varianza2 ** (1/2))
59        print(f"La Varianza Muestral calculada con la formula 1 es: {varianza1M}")
60        print("Su Desviación Estándar es: ", varianza1M ** (1/2))
61        print(f"La Varianza Muestral calculada con la formula 2 es: {varianza2M}")
62        print("Su Desviación Estándar es: ", varianza2M ** (1/2))
63
64    if __name__ == "__main__":
65        main()
```

## Ejecución

```
1 Ingrese la cantidad de datos: 10
2 Dato 1: 16
3 Dato 2: 14
4 Dato 3: 18
5 Dato 4: 15
6 Dato 5: 16
7 Dato 6: 14
8 Dato 7: 16
9 Dato 8: 18
10 Dato 9: 19
11 Dato 10: 15
12 El Promedio es: 16.1
13 La Varianza Poblacional calculada con la formula 1 es: 2.6900000000000004
14 Su Desviación Estándar es: 1.6401219466856727
15 La Varianza Poblacional calculada con la formula 2 es: 2.6900000000000093
16 Su Desviación Estándar es: 1.6401219466856753
17 La Varianza Muestral calculada con la formula 1 es: 2.988888888888889
18 Su Desviación Estándar es: 1.6401219466856727
19 La Varianza Muestral calculada con la formula 2 es: 2.988888888888899
20 Su Desviación Estándar es: 1.6401219466856753
```

### » EJERCICIO 7

Imprimir números del 1 al 10, usando la estructura while.

- Clase: NumeroNatural.
- Atributo: Valor.
- Acción: Mostrar.
- Objeto: numero = NumeroNatural.

## Código

```
1 class NumeroNatural:
2     def __init__(self, valor):
3         self.valor = valor
4
5     def mostrar(self):
6         print(f"Número natural: {self.valor}")
7
8 def main():
9
10     i = 1
11     while i <= 10:
12         numero = NumeroNatural(i)
13         numero.mostrar()
14         i += 1
15
16 if __name__ == "__main__":
17     main()
```

## Ejecución

```
1 Número natural: 1
2 Número natural: 2
3 Número natural: 3
4 Número natural: 4
5 Número natural: 5
6 Número natural: 6
7 Número natural: 7
```

```

8 Número natural: 8
9 Número natural: 9
10 Número natural: 10

```

## » EJERCICIO 8

En una lista del 1 al 10 calcular si es múltiplo de 3, 5 o ninguno.

- Clase: NumeroMultiplo.
- Atributo: Valor.
- Acción: mostrarMultiplo.
- Objeto: numero = NumeroMultiplo().

### Código

```

1 class NumeroMultiplo:
2     def __init__(self, valor):
3         self.valor = valor
4
5     def mostrarMultiplo(self):
6         if self.valor == 0:
7             print(f"El {self.valor} es número nulo")
8         elif self.valor % 3 == 0 and self.valor % 5 == 0:
9             print(f"El {self.valor} es múltiplo de 3 y de 5")
10        elif self.valor % 3 == 0:
11            print(f"El {self.valor} es múltiplo de 3")
12        elif self.valor % 5 == 0:
13            print(f"El {self.valor} es múltiplo de 5")
14        else:
15            print(f"El {self.valor} no es múltiplo de 3 ni de 5")
16
17        print("*"*60)
18
19 def main():
20
21     i = 0
22     while i <= 10:
23         numero = NumeroMultiplo(i)
24         numero.mostrarMultiplo()
25         i += 1
26 if __name__ == "__main__":
27     main()

```

### Ejecución

```

1 El 0 es número nulo
2 *****
3 El 1 no es múltiplo de 3 ni de 5
4 *****
5 El 2 no es múltiplo de 3 ni de 5
6 *****
7 El 3 es múltiplo de 3
8 *****
9 El 4 no es múltiplo de 3 ni de 5
10 *****
11 El 5 es múltiplo de 5
12 *****
13 El 6 es múltiplo de 3
14 *****
15 El 7 no es múltiplo de 3 ni de 5
16 *****

```

```
17 El 8 no es múltiplo de 3 ni de 5
18 *****
19 El 9 es múltiplo de 3
20 *****
21 El 10 es múltiplo de 5
22 *****
```