Universidad Autónoma de Baja California

Alumno: Colin Reyes Brian Gabriel

Matrícula: 379494

Materia: Phyton

Docente: Pedro Nuñez Yepiz

Periodo: Agosto - Diciembre

Grado/semestre: 3er semestre

Grupo: 432

Tema: Actividad 4

Fecha: 09/09/25

Teoría sobre try, for, range() y random en Python

1. try

-Sirve para manejar errores en un programa.
-Permite que el programa no se detenga cuando ocurre un error.
-Siempre se utiliza junto con except.
-Estructura básica:
-try: → bloque donde puede ocurrir el error.
-except: → bloque que se ejecuta si ocurre un error.

2. for

-Es un bucle que permite repetir instrucciones.
-Recorre elementos de una secuencia (lista, cadena, tupla, etc.).
-También se combina con range() para repetir un número determinado de veces.

3. range()

-Es una función que genera una secuencia de números enteros.
-Sintaxis: range(inicio, fin, paso)
-inicio → número desde donde empieza (por defecto 0).
-fin → número hasta donde llega (sin incluir este valor).
-paso → incremento entre números (por defecto 1).

Ejemplos:

-range(5) → 0, 1, 2, 3, 4
-range(2, 10, 2) → 2, 4, 6, 8

4. random

-Es un módulo de Python que permite trabajar con números aleatorios.
-Se debe importar con: import random.
-Funciones más usadas:
-random.random() → número decimal entre 0.0 y 1.0.
-random.randint(a, b) → número entero entre *a* y *b* (incluye ambos).
-random.choice(lista) → selecciona un elemento al azar de una lista.
-random.shuffle(lista) → mezcla aleatoriamente los elementos de una lista.

EJERCICIO 1: Números Aleatorios (Par/Impar)

Generar 40 enteros aleatorios entre 0 y 200, mostrar cada número indicando si es par o impar, contar cuántos pares e impares hay y calcular la suma de los pares y la suma de los impares. Aplicando for, random.randint, el operador %, contadores y acumuladores.

```python
import random
numeros = [random.randint(0, 200) for _ in range(40)]

count_par = 0
count_impar = 0
sum_par = 0
sum_impar = 0

print("Números generados:")
print(numeros)
print("\nClasificación individual:")
print("{:>4}  {:>6}".format("Índice","Valor (tipo)"))
print("-" * 20)

for i, n in enumerate(numeros, start=1):
    tipo = "par" if n % 2 == 0 else "impar"
    print(f"{i:>4}. {n:>4}  ({tipo})")
    if n % 2 == 0:
        count_par += 1
        sum_par += n
    else:
        count_impar += 1
        sum_impar += n

print("\nResumen:")
print(f"Cantidad de pares:   {count_par}")
print(f"Suma de los pares:   {sum_par}")
print(f"Cantidad de impares: {count_impar}")
print(f"Suma de los impares: {sum_impar}")
```

```
Números generados:
[47, 1, 159, 194, 182, 26, 161, 119, 134, 50, 110, 94, 142

Clasificación individual:
Índice  Valor (tipo)
--------------------
   1.   47  (impar)
   2.    1  (impar)
   3.  159  (impar)
   4.  194  (par)
   5.  182  (par)
   6.   26  (par)
   7.  161  (impar)
   8.  119  (impar)
   9.  134  (par)
  10.   50  (par)
  11.  110  (par)
  12.   94  (par)
  13.  142  (par)
  14.   54  (par)
  15.  166  (par)
  16.   34  (par)
  17.  116  (par)
  18.  192  (par)
  19.  157  (impar)
  20.    6  (par)
  21.  113  (impar)
  22.   21  (impar)
  23.    1  (impar)
  24.  169  (impar)
  25.  106  (par)
  26.   65  (impar)
  27.   25  (impar)
  28.  171  (impar)
  29.   31  (impar)
  30.   26  (par)
  31.   14  (par)
  32.  138  (par)
  33.  138  (par)
  34.  108  (par)
  35.  161  (impar)
  36.    5  (impar)
  37.   63  (impar)
  38.   73  (impar)
  39.  176  (par)
  40.  199  (impar)

Resumen:
Cantidad de pares:   21
Suma de los pares:   2206
Cantidad de impares: 19
Suma de los impares: 1741
```

EJERCICIO 2: Tabla de Multiplicar

El programa solicita al usuario un número entero entre 1 y 20, valida que esté dentro del rango y luego despliega la tabla de multiplicar de ese número del 1 al 10 usando un ciclo for.

```python
while True:
    try:
        n = int(input("Ingrese un número entre 1 y 20: "))
        if 1 <= n <= 20:
            break
        print("Número fuera de rango. Intente nuevamente.")
    except ValueError:
        print("Entrada no válida. Ingrese un entero.")

print(f"\nTabla de multiplicar de {n}:")
for i in range(1, 11):
    print(f"{n} * {i} = {n * i}")
```

```
Ingrese un número entre 1 y 20: 2323
Número fuera de rango. Intente nuevamente.
Ingrese un número entre 1 y 20: 4

Tabla de multiplicar de 4:
4 * 1 = 4
4 * 2 = 8
4 * 3 = 12
4 * 4 = 16
4 * 5 = 20
4 * 6 = 24
4 * 7 = 28
4 * 8 = 32
4 * 9 = 36
4 * 10 = 40
```

EJERCICIO 3: Validación de Calificación

El programa solicita una calificación (0–100), maneja errores de captura con try-except, valida el rango y muestra si el alumno está "Aprobado" (≥ 60) o "Reprobado" (< 60).

```python
while True:
    try:
        cal = float(input("Ingrese una calificación (0-100): "))
    except ValueError:
        print("Error de captura: ingrese un número válido.")
        continue

    if not 0 <= cal <= 100:
        print("Calificación fuera de rango. Debe ser entre 0 y 100.")
        continue

    break

estado = "Aprobado" if cal >= 60 else "Reprobado"
print(f"\nCalificación: {cal:.2f} -> {estado}")
```

```
Ingrese una calificación (0-100): 999
Calificación fuera de rango. Debe ser entre 0 y 100.
Ingrese una calificación (0-100): 99

Calificación: 99.00 -> Aprobado
```

EJERCICIO 4: Suma y Media de Números

El programa lee enteros positivos hasta que el usuario ingresa 0. Valida la entrada (entero y no negativo), acumula la suma y cuenta los valores, y al finalizar muestra la suma total y la media. Si no se introdujo ningún número distinto de 0 informa que no hay datos.

```python
total = 0
contador = 0

while True:
    try:
        n = int(input("Ingrese un número entero positivo (0 para terminar): "))
    except ValueError:
        print("Entrada no válida. Ingrese un entero.")
        continue

    if n < 0:
        print("Solo se permiten números positivos. Intente otra vez.")
        continue

    if n == 0:
        break

    total += n
    contador += 1

if contador == 0:
    print("\nNo se ingresaron números.")
else:
    media = total / contador
    print(f"\nSuma total: {total}")
    print(f"Cantidad de números: {contador}")
    print(f"Media: {media:.2f}")6
```

```
Ingrese un número entero positivo (0 para terminar): 1.2
Entrada no válida. Ingrese un entero.
Ingrese un número entero positivo (0 para terminar): -1
Solo se permiten números positivos. Intente otra vez.
Ingrese un número entero positivo (0 para terminar): 4
Ingrese un número entero positivo (0 para terminar): 5
Ingrese un número entero positivo (0 para terminar):
Entrada no válida. Ingrese un entero.
Ingrese un número entero positivo (0 para terminar): 0

Suma total: 9
Cantidad de números: 2
Media: 4.50
```

EJERCICIO 5: Promedio de Materia

El programa permite hasta 3 intentos para ingresar el promedio de una materia (0–100). Valida la entrada con try-except y no cuenta intentos cuando la entrada es inválida. Si el promedio es ≥ 60 muestra felicitaciones y termina; si tras 3 intentos válidos el estudiante sigue reprobando, muestra "Baja Académica".

```python
for intento in range(1, 4):
    while True:
        try:
            prom = float(input(f"Intento {intento}/3 - Ingrese el promedio (0-100): "))
        except ValueError:
            print("Entrada no válida. Ingrese un número.")
            continue

        if not 0 <= prom <= 100:
            print("Fuera de rango. Debe estar entre 0 y 100.")
            continue

        break

    if prom >= 60:
        print(f"\nPromedio {prom:.2f} -> ¡Aprobado! Felicidades, pase al siguiente semestre.")
        break
    else:
        quedan = 3 - intento
        if quedan > 0:
            print(f"Promedio {prom:.2f} -> Reprobado. Quedan {quedan} intento(s).")
        else:
            print(f"\nPromedio {prom:.2f} -> Reprobado.\nBaja Académica")
```

```
Intento 1/3 - Ingrese el promedio (0-100): 111
Fuera de rango. Debe estar entre 0 y 100.
Intento 1/3 - Ingrese el promedio (0-100): 34
Promedio 34.00 -> Reprobado. Quedan 2 intento(s).
Intento 2/3 - Ingrese el promedio (0-100): 54
Promedio 54.00 -> Reprobado. Quedan 1 intento(s).
Intento 3/3 - Ingrese el promedio (0-100): 666
Fuera de rango. Debe estar entre 0 y 100.
Intento 3/3 - Ingrese el promedio (0-100): 76

Promedio 76.00 -> ¡Aprobado! Felicidades, pase al siguiente semestre.
```

EJERCICIO 6: Función para Suma, Media, Mayor y Menor

La función permite leer números hasta que el usuario decida dejar de ingresar más. Valida entradas numéricas, acumula suma y contador, y mantiene el valor mayor y menor. Al finalizar muestra la suma, la media, el mayor y el menor. La función devuelve un diccionario con esos resultados.

```python
def procesar_numeros():
    total = 0.0
    cantidad = 0
    mayor = None
    menor = None

    while True:
        entrada = input("Ingrese un número (o escriba 'salir' para terminar): ").strip()
        if entrada.lower() in ("salir", "fin", "s"):
            break

        try:
            num = float(entrada)
        except ValueError:
            print("Entrada no válida. Ingrese un número o 'salir' para terminar.")
            continue

        total += num
        cantidad += 1
        if mayor is None or num > mayor:
            mayor = num
        if menor is None or num < menor:
            menor = num

        seguir = input("¿Desea ingresar otro número? (s/n): ").strip().lower()
        if seguir == "n":
            break

    if cantidad == 0:
        print("\nNo se ingresaron números.")
        return {"suma": 0.0, "media": None, "mayor": None, "menor": None, "cantidad": 0}

    media = total / cantidad
    print("\nResultados:")
    print(f"Suma:   {total}")
    print(f"Media:  {media}")
    print(f"Mayor:  {mayor}")
    print(f"Menor:  {menor}")
    print(f"Cantidad de números: {cantidad}")

    return {"suma": total, "media": media, "mayor": mayor, "menor": menor, "cantidad": cantidad}

resultados = procesar_numeros()
```

```
Ingrese un número (o escriba 'salir' para terminar): d
Entrada no válida. Ingrese un número o 'salir' para terminar.
Ingrese un número (o escriba 'salir' para terminar): 4
¿Desea ingresar otro número? (s/n): s
Ingrese un número (o escriba 'salir' para terminar): 56
¿Desea ingresar otro número? (s/n): n

Resultados:
Suma:   60.0
Media:  30.0
Mayor:  56.0
Menor:  4.0
Cantidad de números: 2
```

EJERCICIO 7: Generación de Números Impares

La función genera números aleatorios entre 10 y 60 usando un for (hasta 25 intentos) y se detiene si logra 15 números impares antes de llegar al máximo. Clasifica cada número como par o impar, acumula y cuenta ambos grupos, y al final muestra la lista de números generados, la cantidad y la media de pares e impares (si existen).

```python
import random

def generar_impares_y_promedios():

    generados = []
    pares = []
    impares = []
    max_intentos = 25
    objetivo_impares = 15

    for _ in range(max_intentos):
        n = random.randint(10, 60)
        generados.append(n)
        if n % 2 == 0:
            pares.append(n)
        else:
            impares.append(n)

        if len(impares) >= objetivo_impares:
            break

    print("Números generados:")
    print(generados)
    print()

    print(f"Impares ({len(impares)}): {impares}")
    print(f"Pares   ({len(pares)}): {pares}")
    print()
```

```python
    def media(lista):
        return sum(lista) / len(lista) if lista else None

    media_impares = media(impares)
    media_pares = media(pares)

    if media_impares is not None:
        print(f"Media de impares: {media_impares:.2f}")
    else:
        print("No se generaron impares.")

    if media_pares is not None:
        print(f"Media de pares:   {media_pares:.2f}")
    else:
        print("No se generaron pares.")

    return {
        "generados": generados,
        "impares": impares,
        "pares": pares,
        "media_impares": media_impares,
        "media_pares": media_pares
    }

result = generar_impares_y_promedios()
```

```
Números generados:
[58, 35, 27, 50, 46, 11, 53, 24, 17, 43, 56, 28, 39, 26, 46, 58, 41, 15, 20, 51, 55, 36, 49, 53, 42]

Impares (13): [35, 27, 11, 53, 17, 43, 39, 41, 15, 51, 55, 49, 53]
Pares   (12): [58, 50, 46, 24, 56, 28, 26, 46, 58, 20, 36, 42]

Media de impares: 37.62
Media de pares:   40.83
```

EJERCICIO 8: Validación de Número en Rango

La función solicita al usuario un rango (mínimo y máximo) y luego un número para validar si está dentro de ese rango. Repite el proceso tantas veces como el usuario quiera. Al finalizar muestra la cantidad de números válidos ingresados y su promedio. Usa try-except para validar entradas y un while para repetir.

```python
def validar_numeros_en_rango():
    cantidad = 0
    suma = 0.0

    while True:
        while True:
            try:
                minimo = float(input("Ingrese el valor mínimo del rango: "))
                maximo = float(input("Ingrese el valor máximo del rango: "))
            except ValueError:
                print("Entrada no válida. Ingrese números para el rango.")
                continue

            if minimo > maximo:
                print("El mínimo no puede ser mayor que el máximo. Intente de nuevo.")
                continue
            break

        while True:
            try:
                num = float(input(f"Ingrese un número entre {minimo} y {maximo}: "))
            except ValueError:
                print("Entrada no válida. Ingrese un número.")
                continue

            if num < minimo or num > maximo:
```

```python
            if num < minimo or num > maximo:
                print("Número fuera del rango. Intente nuevamente.")
                continue
            break

        cantidad += 1
        suma += num
        print(f"Número aceptado: {num}")

        resp = input("¿Desea validar otro número/rango? (s/n): ").strip().lower()
        if resp == "n":
            break

    if cantidad == 0:
        print("\nNo se ingresaron números válidos.")
        return {"cantidad": 0, "promedio": None}
    promedio = suma / cantidad
    print("\nResultados finales:")
    print(f"Cantidad de números válidos: {cantidad}")
    print(f"Promedio: {promedio:.2f}")

    return {"cantidad": cantidad, "promedio": promedio}

resultados = validar_numeros_en_rango()
```

```
Ingrese el valor mínimo del rango: 4
Ingrese el valor máximo del rango: 44
Ingrese un número entre 4.0 y 44.0: 4
Número aceptado: 4.0
¿Desea validar otro número/rango? (s/n): s
Ingrese el valor mínimo del rango: 4
Ingrese el valor máximo del rango: 44
Ingrese un número entre 4.0 y 44.0: d
Entrada no válida. Ingrese un número.
Ingrese un número entre 4.0 y 44.0: 2
Número fuera del rango. Intente nuevamente.
Ingrese un número entre 4.0 y 44.0: 5
Número aceptado: 5.0
¿Desea validar otro número/rango? (s/n): n

Resultados finales:
Cantidad de números válidos: 2
Promedio: 4.50
```

EJERCICIO 9: Área de un Triángulo

Se define una función area_triangulo(base, altura) que recibe la base y la altura, calcula el área con la fórmula (base * altura) / 2 y devuelve el resultado. El programa principal solicita (o asigna) los valores, valida entradas numéricas positivas y muestra el área.

```python
def area_triangulo(base, altura):
    return (base * altura) / 2

try:
    b = float(input("Ingrese la base del triángulo: "))
    h = float(input("Ingrese la altura del triángulo: "))
except ValueError:
    print("Entrada no válida. Ingrese números.")
else:
    if b <= 0 or h <= 0:
        print("La base y la altura deben ser mayores que 0.")
    else:
        area = area_triangulo(b, h)
        print(f"Área del triángulo: {area:.2f}")
```

```
Ingrese la base del triángulo: 44
Ingrese la altura del triángulo: 44
Área del triángulo: 968.00
```

EJERCICIO 10: Validación de Número en Rango

La función solicita un rango (mínimo y máximo) y luego un número; valida las entradas con try-except y verifica si el número está dentro del rango. Muestra un mensaje indicando si el número es válido o no válido.

```python
def validar_en_rango():
    try:
        minimo = float(input("Ingrese el valor mínimo del rango: "))
        maximo = float(input("Ingrese el valor máximo del rango: "))
    except ValueError:
        print("Entrada no válida. Debe ingresar números para el rango.")
        return

    if minimo > maximo:
        print("Rango inválido: el mínimo es mayor que el máximo.")
        return

    try:
        num = float(input(f"Ingrese un número entre {minimo} y {maximo}: "))
    except ValueError:
        print("Entrada no válida. Debe ingresar un número.")
        return

    if minimo <= num <= maximo:
        print(f"Número {num} -> VÁLIDO dentro del rango [{minimo}, {maximo}].")
    else:
        print(f"Número {num} -> NO VÁLIDO. Está fuera del rango [{minimo}, {maximo}].")

validar_en_rango()
```

```
Ingrese el valor mínimo del rango: 4
Ingrese el valor máximo del rango: 44
Ingrese un número entre 4.0 y 44.0: 4
Número 4.0 -> VÁLIDO dentro del rango [4.0, 44.0].
```

CONCLUSIÓN

Se aplicaron los fundamentos básicos de programación en Python mediante ciclos, condicionales, validación de entradas, manejo de excepciones y funciones personalizadas, logrando resolver problemas prácticos como cálculos matemáticos, clasificación de datos y validación de rangos. Estos ejercicios reforzaron la lógica de programación y el uso de estructuras esenciales del lenguaje.

GITHUB

https://github.com/resu0044/Phyton