|  |  |  |
| --- | --- | --- |
| | GUÍA 3.4.2:Guía Ejercicios Resueltos Archivos | | |
| Sigla | Asignatura | Experiencia de Aprendizaje |
| FPY1101 | Fundamentos de Programación | EA2: Optimizando el código en Python |
| Tiempo | Modalidad de Trabajo | Indicadores de logro |
| 2 h | Individual | IL 3.1 al IL 3.4 |

|  |
| --- |
| **Código QR con relleno sólido**  **Antecedentes generales** |

## Esta guía tiene como objetivo conocer los aspectos generales en Python

### Ser una guía paso a paso para la construcción de programas en Python

### Todos los ejercicios tienen la solución incluida, pero antes de ver la solución, debes resolver por tu cuenta el ejercicio, de esa manera podrás reforzar y aprender. Las soluciones utilízalas para comparar con tus resultados, tomar nota o simplemente como revisión. debes ser consiente y responsable en tu autoaprendizaje.

### Debate con tu docente las respuestas obtenidas, si tienes dudas, recuerda dar aviso y argumentar, los programas no tienen soluciones únicas, teniendo derivaciones o mecanismos distintos de funcionamiento.

### Esta guía puede ser desarrollada en casa, o guiada por el docente, con el fin de seguir un paso a paso y comprender las explicaciones de la o el docente.

|  |
| --- |
| **Lista con relleno sólido Requerimientos para esta actividad** |

## Para el desarrollo de esta actividad deberás disponer de:

## Computador

## Visual Studio Code

|  |  |
| --- | --- |
| **Inteligencia artificial con relleno sólidoActividad** |  |

**EJERCICIO 1 Calculadora con funciones**

Creen un programa que emule la función de una calculadora, debe tener 4 funciones, sumar, restar, dividir y multiplicar, el programa de permitir el ingreso de dos números de tipo enteros. Este programa debe contener una pequeña validación que indique un mensaje cuando se divide por 0, indicado que no se puede realizar la operación. Las funciones a construir deben ser con argumentos y con retorno.

Solución

def sumar(a, b):

return a + b

def restar(a, b):

return a - b

def multiplicar(a, b):

return a \* b

def dividir(a, b):

if b == 0:

return "Error: No se puede dividir por cero."

else:

return a / b

num1 = int(input("Ingrese el primer número: "))

num2 = int(input("Ingrese el segundo número: "))

resultado\_suma = sumar(num1, num2)

resultado\_resta = restar(num1, num2)

resultado\_multiplicacion = multiplicar(num1, num2)

resultado\_division = dividir(num1, num2)

print(f"Suma: {resultado\_suma}")

print(f"Resta: {resultado\_resta}")

print(f"Multiplicación: {resultado\_multiplicacion}")

print(f"División: {resultado\_division}")

**EJERCICIO 2 Incorporación de sentencias de validación**

Al ejercicio anterior, le debemos incorporar una función de validación, que permita validar que solo se puede ingresar números. Además, el ejercicio anterior no permite ingresar valores decimales, a lo cual, en la misma función de validación se debe incorporar una conversión de valor de tipo Int a valor de tipo Float

Solución:

def sumar(a, b):

return a + b

def restar(a, b):

return a - b

def multiplicar(a, b):

return a \* b

def dividir(a, b):

if b == 0:

return "Error: No se puede dividir por cero."

else:

return a / b

def validar\_num(numero\_entrada):

while True:

try:

numero = float(input(numero\_entrada))

return numero

except ValueError:

print("Error: Ingrese un valor numérico válido.")

# Ingreso de dos números con validaciones

num1 = validar\_num("Ingrese el primer número: ")

num2 = validar\_num("Ingrese el segundo número: ")

resultado\_suma = sumar(num1, num2)

resultado\_resta = restar(num1, num2)

resultado\_multiplicacion = multiplicar(num1, num2)

resultado\_division = dividir(num1, num2)

print(f"Suma: {resultado\_suma}")

print(f"Resta: {resultado\_resta}")

print(f"Multiplicación: {resultado\_multiplicacion}")

print(f"División: {resultado\_division}")