Nombre: Miguel Alejandro Lomeli Haro

Materia: Computación tolerante a fallas

Actividad: Otras herramientas para el manejar errores parte 2

Introducción:
Con la realización de esta práctica se generará un código aplicando alguna de las herramientas que se investigaron en la tarea parte 1, donde se mostrará como funciona es parte del código y como el programa continúa a pesar del error.

Contenido:

```python
def division_segura(a, b):
    try:
        resultado = a / b
        return resultado
    except ZeroDivisionError:
        return "Error: División entre cero no es valida"
    except TypeError:
        return "Error: Tipos de datos no válidos para la operación"

# Ejemplos de uso
num1 = 10
num2 = 0
resultado1 = division_segura(num1, num2)
print("\n",resultado1)

num3 = "10"
num4 = 2
resultado2 = division_segura(num3, num4)
print("\n",resultado2)

num5 = 10
num6 = 2
resultado3 = division_segura(num5, num6)
print("\n",resultado3,"\n")
```

En este programa se utiliza el try catch donde es una división, la cual si es entre 0 proporciona un mensaje de error, de igual manera si es una división con tipos de datos no válidos aparecerá otro error.

```
Error: División entre cero no es valida

Error: Tipos de datos no válidos para la operación

5.0
```

Aquí se pueden ver los resultados de las operaciones de las divisiones.

Conclusiones:
Con la realización de esta actividad, puse en práctica lo aprendido en la primera parte de esta actividad que fue la investigación de herramientas para el manejo de errores, y pude aplicarlo de manera correcta ya que el programa funcionó de manera correcta a pesar de generar errores.

Bibliografía:

- Kamunya, T. (2023). Las 11 mejores herramientas de seguimiento de errores para equipos de desarrollo modernos. *Geekflare*. https://geekflare.com/es/bug-tracking-tools/
- *8. Errores y excepciones*. (s. f.). Python documentation. https://docs.python.org/es/3/tutorial/errors.html