MANEJO DE ERRORES EN PROGRAMACION

Emanuel Alejandro Gutierrez Romero

28/08/23

COMPUTACION TOLERANTE A FALLAS

Prof. Michel Emanuel Lopez Franco

Introducción

Durante el desarrollo de software hay que tener en cuentas varias cosas sobre el usuario, la primera es que no siempre va a hacer lo que queramos que haga y a veces solo va al programa a ver como se pude romper, para esto tenemos que adelantarnos a esto. Un ejemplo de esto es la división entre 0 y ahora veremos cómo resolverlo.

Desarrollo

En una función de división como la siguiente.

```
def division(num,div):
 return num/div
```

Podemos ingresar en los números que queramos y funcionara mientras se ingresen números que se puedan dividir, pero cuando ingrese 0 en `div` Es cuando los problemas empiezan, por ejemplo.

```
division(27,0)
```

Saldrá lo siguiente.

```
ZeroDivisionError                         Traceback (most recent call last)
<ipython-input-19-932ea024ce43> in <module>()
----> 1 dividir(27,0)

<ipython-input-1-c98670fd7a12> in dividir(num, div)
      1 def divide(num,div):
----> 2   return num/div

ZeroDivisionError: division by zero
```

Lo que indica que como en la vida real, la división entre 0 no esta definida.

Pero para resolver este problema tenemos lo que se conoce como Try y Except este tras poner la línea de `ZeroDivisionError` podemos mandarle un mensaje de error al usuario, por ejemplo.

```
try:
    res = division(num, div)
    print(res)
except ZeroDivisionError:
    print("Error dividir entre 0 no es posible... ")
```

Con esto haremos saber al usuario que no es posible dividir entre 0 sin embargo podemos manejar esto de otra forma como en el siguiente código.

```
def division(a, b):
    if b != 0:
        res = a / b
        return res
    else:
        print("Error: No se puede dividir entre 0.")
        return None

num = float(input("Ingrese el numero: "))
div = float(input("Ingrese el divisor: "))

res = division(num, div)

if res is not None:
    print("El resultado de la división es:", res)
```

El cual sin un try except puede manejar el error de la división entre 0.

Conclusiones

Considero que el detalle de la división entre 0 es importante ya que puedes poner un mensaje como recordatorio tipo “Hola usuario solo te recuerdo que la división entre 0 no está definida, Gracias por no romper el programa que tanto me costó hacer :)”, sin embargo y siendo honestos habrá usuarios que, aunque se les advierta lo harán, por eso es mejor prevenir y estar consientes de que va a suceder y saber como manejarlo, a simplemente esperar que el usuario se porte bien y no rompa nada.

Bibliografía

- https://www.freecodecamp.org/espanol/news/sentencias-try-y-except-de-python-como-menejar-excepciones-en-python.