MANEJO DE EXCEPCIONES

CURSO PYTHON

CILSA

Estructura de control Try Except

Por último, vamos a ver otra estructura, que no es iterativa. Es una estructura o bloque muy utilizada en programas de grandes magnitudes, el **Try Except**.

Esta estructura es la conocida como “capturadora de excepciones”. Su funcionalidad es la de si pasa algo que detiene al flujo (un error) pueda capturar ese error, mostrarnos algún mensaje relacionado con el mismo, pero no corta la ejecución del programa, sino que deja que continúe.

Esto sirve mucho cuando tenemos una parte del código que no es tan crítica, pero sí necesitamos que el programa continúe con su ejecución, como puede ser una conexión a una base de datos, un error de un servicio, algún error de entrada, etc.

Excepciones

Los nombres de las excepciones son en inglés. Es una frase donde cada palabra comienza con mayúscula, y toda la frase va unida, sin espacios.

El editor de código generalmente sugiere como primera opción el autocompletado al escribir la primera parte de la frase.

Control de excepciones

Para controlar las excepciones usamos las palabras reservada **TRY** y **EXCEPT**.
Dentro del bloque de código del **TRY** (luego de los dos puntos) colocamos toda nuestra lógica de ejecución, todo lo que queremos hacer.
Luego, "**a la altura**" del **TRY**, colocamos **EXCEPT** seguido del dos puntos. A partir del siguiente renglón estaremos en el bloque de código del mismo, el cual se ejecutará cuando ocurra una excepción en la ejecución del programa o Script.
Entonces tenemos la siguiente estructura:

try:
    bloque código del TRY
except:
    bloque código del EXCEPT

Sintaxis del Try Except

Su forma de utilización es la siguiente:

```
try:
    #codigo a ejecutar donde puede ocurrir un error
    a = 5
    b = 0
    division = a/b
    print(division)
except:
    print(“Hubo un problema”)
```

Aquí lo que sucede, es que como no se puede dividir por 0, nuestro programa automáticamente se frena y nos saltará un error por pantalla por defecto de Python. Lo que hacemos es decirle a Python, que está bien, sabemos que puede ocurrir un error, pero que solo lo vamos a capturar, dar el mensaje “hubo un problema” y seguiremos con nuestro programa.

Capturar errores

Lo que nos permite también el except, es que si conocemos de qué tipo es el error, podremos capturarlo y renombrarlo para que nos dé mejor información de lo que está pasando:

print(x) #nos da un error de “NameError: name ‘x' is not defined” qué es que “x” no está definida.

Para solucionar esto y que se entienda mejor el tipo de error, podríamos capturar este error:

```
try:
        print(x)
except NameError:
        print(“La variable que trata de mostrar por pantalla no está definida” )
```

Entonces ahora podremos entender mejor porque fue el error.

Tipos de errores

Los nombres de las excepciones y su descripción son las siguientes:

- TypeError: Ocurre cuando se aplica una operación o función a un dato del tipo inapropiado.
- ZeroDivisionError: Ocurre cuando se intenta dividir por cero.
- OverflowError: Ocurre cuando un cálculo excede el límite para un tipo de dato numérico.
- IndexError: Ocurre cuando se intenta acceder a una secuencia con un índice que no existe.
- KeyError: Ocurre cuando se intenta acceder a un diccionario con una clave que no existe.
- FileNotFoundError: Ocurre cuando se intenta acceder a un archivo que no existe en la ruta indicada.
- ImportError: Ocurre cuando falla la importación de un módulo.

Sentencias opcionales

También podemos incluir sentencias opcionales dentro del bloque de excepciones: **ELSE** y **FINALLY**.

Ambos se colocan “**a la altura**” o “**al nivel**” del TRY y EXCEPT.

ELSE: podemos colocar instrucciones para que se ejecute si todo salió bien, es decir, cuando se ejecutó de manera correcta el bloque TRY.

FINALLY: para instrucciones que siempre van a ejecutarse, sin importar si se ejecutó el bloque TRY de manera correcta o si hubo alguna excepción.

Ejemplo de sentencias opcionales

```
try:
    divisor = int( input( “ingrese un divisor: ”))
    x = 100 / divisor
except:
    bloque código del EXCEPT
else:
    print(“No ocurrió ninguna excepción”)
finally:
    print(“Siempre me ejecuto”)
```

Conclusión

Con estas estructuras podremos realizar verificaciones, automatizar procesos, y capturar errores.

Cada vez nos estamos acercando a crear un programa que pueda resolver cualquier problema que le planteemos siempre y cuando esté a su alcance.

RECUERDEN QUE LA MÁQUINA HACE LO QUE LE DECIMOS, ¡NO HACE MAGIA!