Day 8 – Exception Handling

● When an error occurs, or exception as we call it, Python will normally stop and generate an error message. These exceptions can be handled using the try statement:

In [1]:
```
try:
    print(a)
except:
    print("An exception occurred")
```

```
An exception occurred
```

● If you want to execute a special block of code for a special kind of error like the if else block we can specify the error.

In [2]:
```
try:
    print(player)
except NameError:
      print("player not defined")
except:
      print("End of program")
```

```
player not defined
```

● Use the else keyword to define a block of code to be executed if no errors were raised

In [3]:
```
try:
    print(x)
except:
    print('Exception')
else:
    print('End of code')
```

```
Exception
```

● Finally block, if specified, will be executed regardless if the try block raises an error or not

In [4]:
```
try:
    print(x)
except:
    print("Something went wrong")
finally:
    print("The 'try except' is finished")
```

```
Something went wrong
The 'try except' is finished
```

Raise an exception

- As a Python developer you can choose to throw an exception if a condition occurs.
- To throw (or raise) an exception, use the raise keyword.

In [5]:
```
x = -1
if x < 0:
    raise Exception('Sorry, no numbers below zero')
```

```
---------------------------------------------------------------------------
Exception                                 Traceback (most recent call last)
<ipython-input-5-95c31ad72c04> in <module>
      1 x = -1
      2 if x < 0:
----> 3     raise Exception('Sorry, no numbers below zero')

Exception: Sorry, no numbers below zero
```

Exercises :

1.List down all the error types and check all the errors using a python program for all errors

ZeroDivisionError

In [6]:
```
try:
    print(10 / 0)
except ZeroDivisionError:
    print("Unable to Divide By Zero")
```

```
Unable to Divide By Zero
```

KeyError

In [7]:
```
dicti = {"1" : 1, "2" : 2, "3" : 3}

try:
    print(dicti["4"])
except KeyError:
    print('Key not found in dictionary')
```

```
Key not found in dictionary
```

IndexError

In [8]:
```
arr = [1, 2, 3]

try:
    print(arr[3])
except IndexError:
    print('Index not found in array')
```

```
Index not found in array
```

ModuleNotFoundError

In [9]:
```
try:
    import flask
except ModuleNotFoundError:
    print("Module not found")
```

2. Design a simple calculator app with try and except for all use cases

In [11]:
```
symbols = '+ - x / % \n'

try:
    input_one = int(input('Enter your 1st input :'))
    print(symbols)
    chosen_symbol = input('Choose your symbol from above :')
    input_two = int(input('Enter your 2st input :'))

    if chosen_symbol in symbols:
        if chosen_symbol == '+':
            print(input_one, '+', input_two, '=',input_one + input_two)
        elif chosen_symbol == '-':
            print(input_one, '-', input_two, '=',input_one - input_two)
        elif chosen_symbol == 'x':
            print(input_one, 'x', input_two, '=',input_one * input_two)
        elif chosen_symbol == '/':
            print(input_one, '/', input_two, '=',input_one / input_two)
        elif chosen_symbol == '%':
            print(input_one, '%', input_two, '=',input_one % input_two)
except ValueError:
    print("Enter Proper Numbers For Input!")
except ZeroDivisionError:
    print("Unable to Divide by Zero (0) !")
```

```
Enter your 1st input :25
+ - x / %

Choose your symbol from above :%
Enter your 2st input :4
25 % 4 = 1
```

3. print one message if the try block raises a NameError and another for other errors

In [13]:
```
try:
    print(x)
except NameError:
    print("You should define variable.")
```

```
-1
```

4. When try-except scenario is not required?

try-except scenario is not required if you are not going to have any runtime error in your python program. if there is any possibility that there might exist a runtime error, you must use a try - except scenario in order to avoid a crash, and guide the user with proper message.

5. Try getting an input inside the try catch block

In [14]:
```
try:
    num = int(input("Enter a number: "))
except ValueError:
    print("Invalid Input")
```

```
Enter a number: g
Invalid Input
```

Completed Day 8's notes & exercises

THANK YOU!

Check out My Repository at https://github.com/AakankshaJarode/BestEnlist_Python_Internship.git (https://github.com/AakankshaJarode/BestEnlist_Python_Internship.git)

Chech out My LinkedIn Page at https://www.linkedin.com/in/aakanksha-jarode-1b0195179 (https://www.linkedin.com/in/aakanksha-jarode-1b0195179)

In [ ]: