Week-9 **Try / Except Exception Handling**

Computational Mathematics (4MM013)

When an error occurs, or exception as we call it, Python will normally stop and generate an error message.

The ***try*** block lets you test a block of code for errors.

The ***except*** block lets you handle the error.

The ***else*** block lets you execute code when there is no error.

The ***finally*** block lets you execute code, regardless of the result of the try- and except blocks.

```
#Program to Divide a Simple Number
number1 = int(input("Enter a number: "))
number2 = int(input("Enter another number: "))
result = number1 / number2
print("The result is:", result)
```

```
Enter a number: 69
Enter another number: 90
The result is: 0.7666666666666667
```

Q1. Try divide with terms 1/2, 1/0 and 1/n in above Program.What are the type of errors you see?

Ans:

```
#Program to solve the above
try:
  number1 = int(input("Enter a number: "))
  number2 = int(input("Enter another number: "))
  result = number1 / number2
  print("The result is:", result)
except ZeroDivisionError:
  print("Error: Cannot divide by zero.")
except ValueError:
  print("Error: Invalid input. Please enter a number.")
```

```
Enter a number: 21
Enter another number: 0
Error: Cannot divide by zero.
```

Example Code:

Try to open and write to a file that is not writable:

```
try:
  f = open("demofile.txt")
  try:
    f.write("Lorum Ipsum")
  except:
    print("Something went wrong when writing to the file")
  finally:
    f.close()
except:
  print("Something went wrong when opening the file")
```

```
Something went wrong when opening the file
```

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.

```
#Raise an error and stop the program if x is lower than 0:

x = -1

if x < 0:
  raise Exception("Sorry, no numbers below zero")
```

```
#Raise a TypeError if x is not an integer:

x = "hello"

if not type(x) is int:
  raise TypeError("Only integers are allowed")
```

Task 1:

Create a py program that will ask for two int values from the user; value A and B.

Have the program return the outcome of dividing A by B. The program should use try / except blocks to deal with divide by zero exceptions, and also invalid data entry or 'Value errors' for A and B.

```
try:
  a = int(input("Enter a number "))
  b = int(input("Enter sec number "))
  c = a/b
  #prints the result if nothing is wrong.
  print("the result is " , c)
  # only if the values are except any integers.
except ValueError:
  print("invalid value!")
  # if the value is divied by 0.
except ZeroDivisionError:
  print("the answer is infinte.")
```

```
    Enter a number 3
    Enter sec number g
    invalid value!
```

Task 2:

Create a program that has a list of 6 numbers and asks for an index as input from the user. The program should print the value at that index in the list, as well as handle exceptions for data type and also 'indexError' for an index higher than the size of the list.

```
numbers = [2,3,4,1,2,69]
try:
  values = int(input(" enter a value btwn 0 to 5 "))
  print("the value is", numbers[values])
except IndexError:
  print(" value needs to be btwn 0 to 5. ")
except ValueError:
  print ("invalid value")
```

```
     enter a value btwn 0 to 5 3
    the value is 1
```

Colab paid products - Cancel contracts here

✓ 4s completed at 1:25 AM