**DEPARTMENT OF INFORMATION TECHNOLOGY**

**COURSE CODE:** DJ19ITL406

**COURSE NAME:** Programing Laboratory 2 (Python)  **CLASS:** SYBTECH

**EXPERIMENT NO. 7**

**CO/LO: CO1, CO2.**

**AIM / OBJECTIVE:**

Write python programs to implement Exception handling.

**DESCRIPTION OF EXPERIMENT:**

1. Try-except-finally
2. Assert
3. User defined exceptions

**QUESTIONS:**

1. Make your exception class “Invalid Marks” which is thrown when marks obtained by student exceeds100

class InvalidMarks(Exception):

  def \_\_init\_\_(self, message):

    self.message = message

try:

    marks=int(input("enter your marks "))

    if(marks>100):

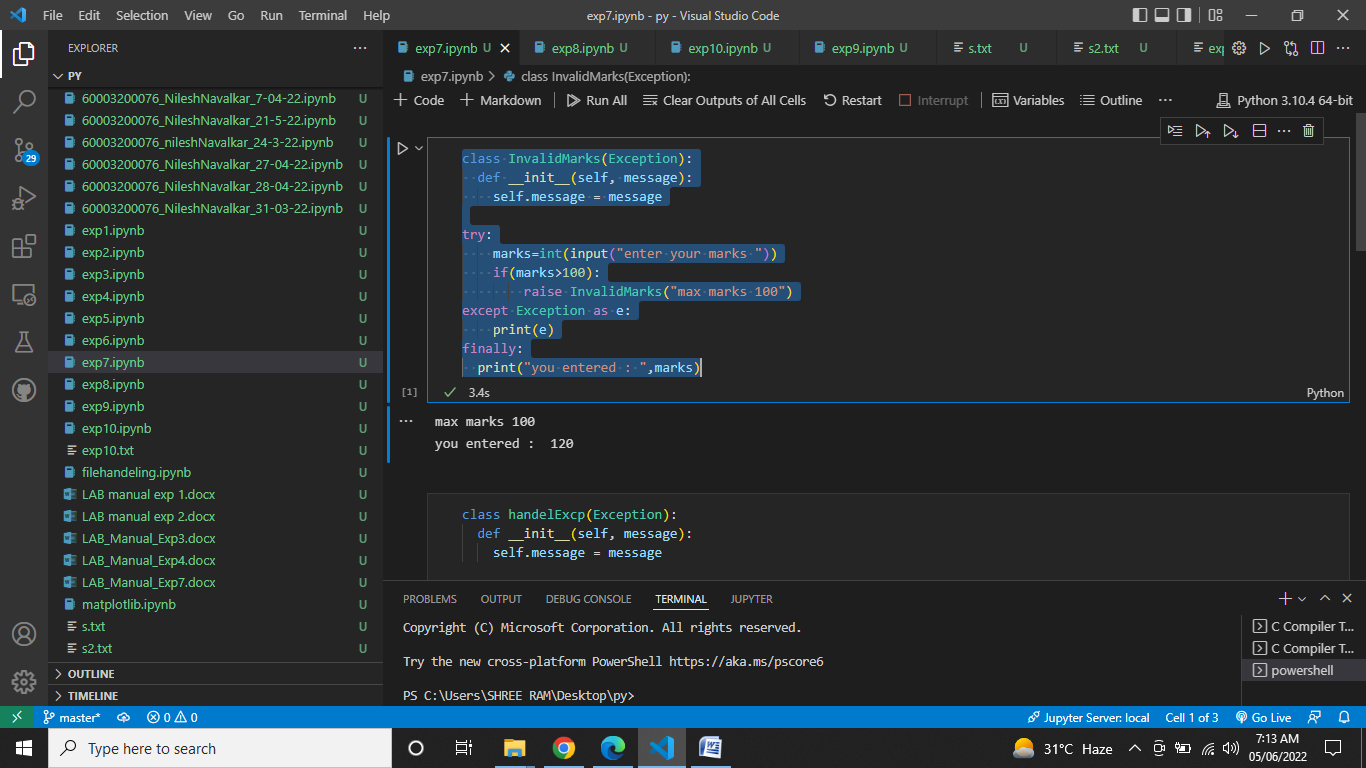
        raise InvalidMarks("max marks 100")

except Exception as e:

    print(e)

finally:

  print("you entered : ",marks)



1. WAP that accepts the values of a,b, c and d.Calculateanddisplay((a+d)+(b\*c))/(b\*d).
2. Create user defined exception to display proper message when value of (b\*d) is zero

class handelExcp(Exception):

  def \_\_init\_\_(self, message):

    self.message = message

a=int(input())

b=int(input())

c=int(input())

d=int(input())

print("A =",a,"  B =",b,"  C =",c,"  D =",d)

try:

  if(b==0 or d==0):

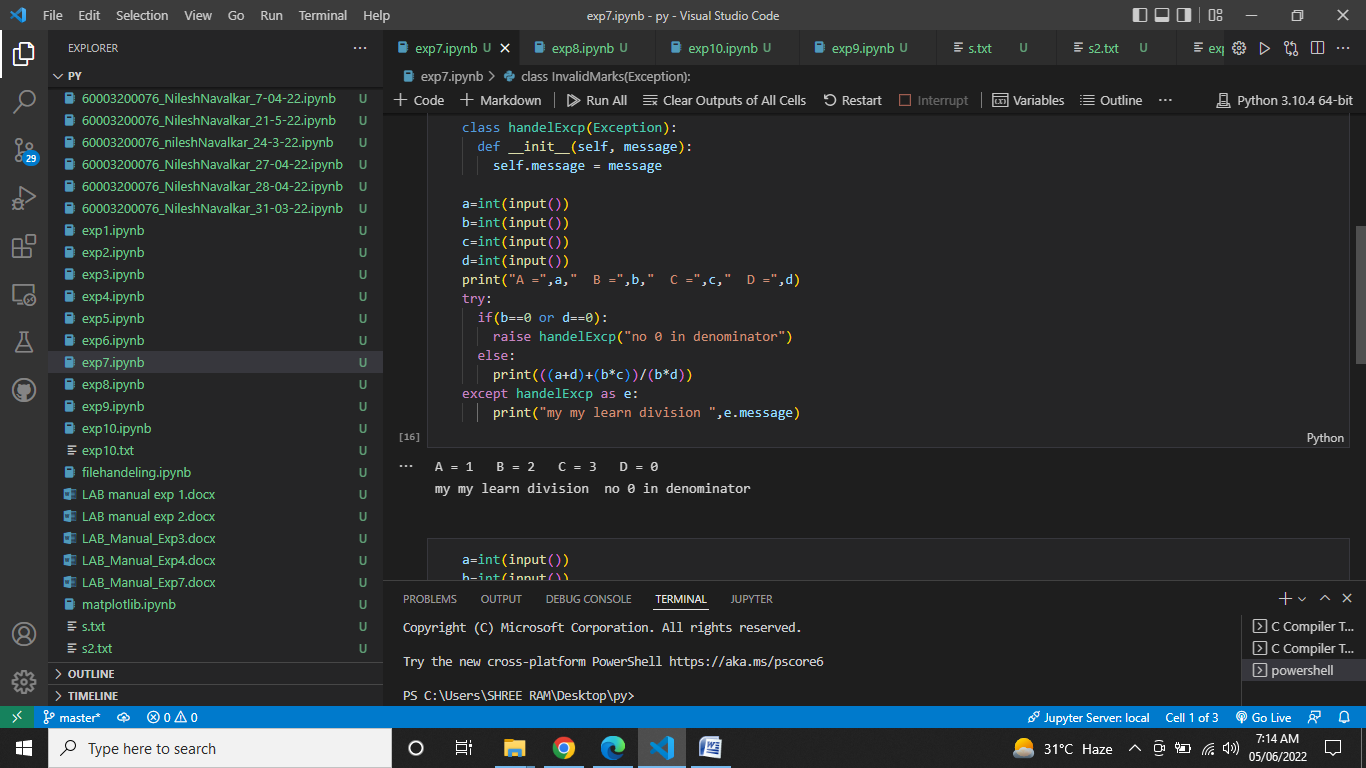
    raise handelExcp("no 0 in denominator")

  else:

    print(((a+d)+(b\*c))/(b\*d))

except handelExcp as e:

    print("my my learn division ",e.message)



1. Make use of assert statement to catch Assertion Error

a=int(input())

b=int(input())

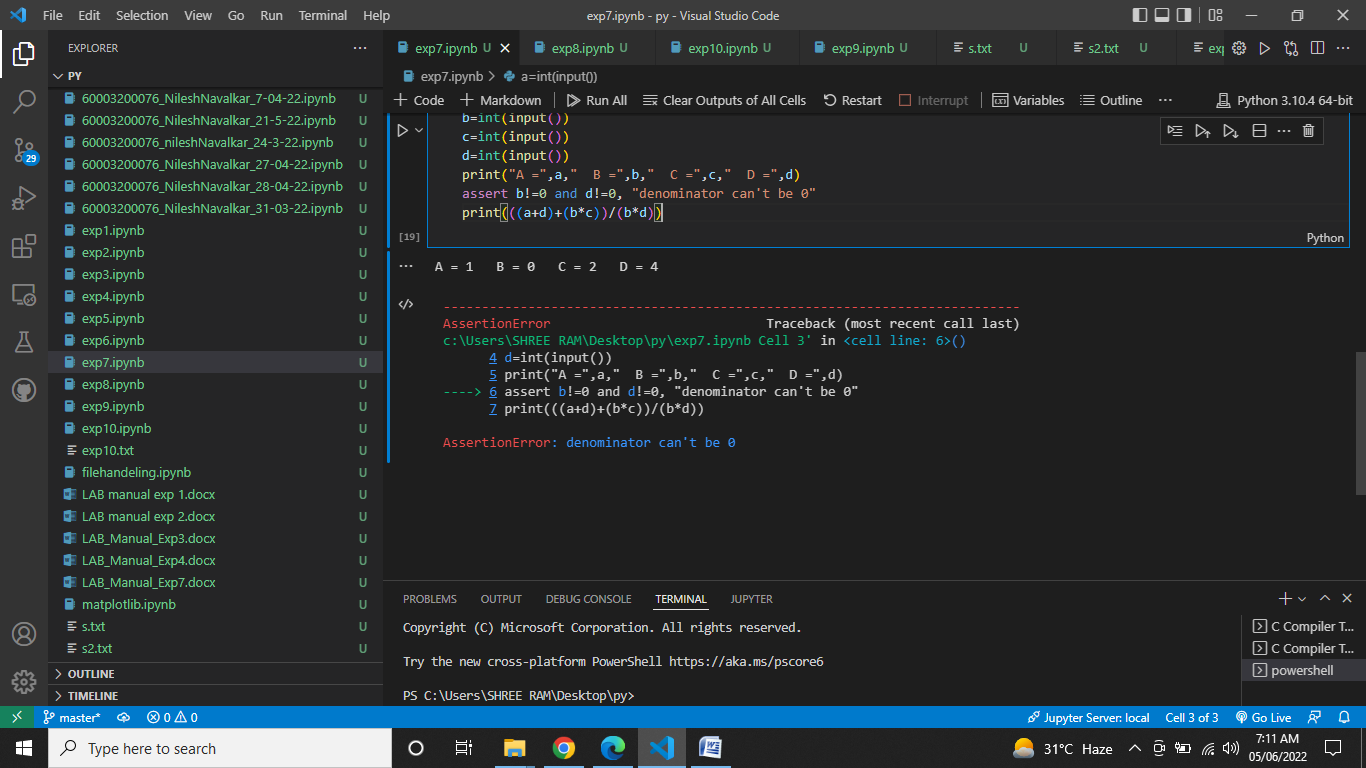
c=int(input())

d=int(input())

print("A =",a,"  B =",b,"  C =",c,"  D =",d)

assert b!=0 and d!=0, "denominator can't be 0"

print(((a+d)+(b\*c))/(b\*d))



**OBSERVATIONS / DISCUSSION OF RESULT:**

We saw above by using try, except, raise, assert, etc. we were able to handle exceptions and even make our own exceptions. The results showed us that python provides a way to handle the exception so that the code can be executed without any interruption. If we do not handle the exception, the interpreter doesn't execute all the code that exists after the exception.

**CONCLUSION:**

Thus, this experiment demonstrated the implementation of try, except, assert, etc. for our programs to run without interruption. Python has many built-in exceptions that enable our program to run without interruption and give the output. We can also define our own exceptions.

**REFERENCES:**

**Website References:​**

[1] https://www.w3schools.com/python