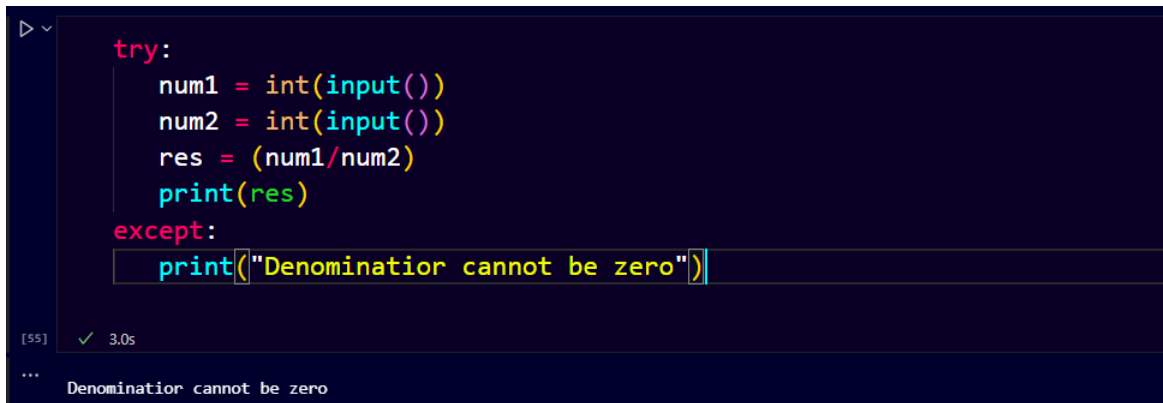


1. What is the role of try and exception block?

Solution:- To handle the error during the execution of the code if we do not use the try and catch block and if error comes then the execution of programme stop and if we use try and catch then it will execute the catch block and the execution will not stop

2. What is the syntax for a basic try-except block?

Solution:-



```
try:
    num1 = int(input())
    num2 = int(input())
    res = (num1/num2)
    print(res)
except:
    print("Denominator cannot be zero")
```

[55] ✓ 3.0s

... Denominator cannot be zero

3. What happens if an exception occurs inside a try block and there is no matching except block?

Solution:- It will give the error which is “incomplete input”

```
num1 = int(input())
num2 = int(input())
try:
    res = (num1/num2)
    print(res)
```

⊗ 0.0s

Cell In[6], line 5

```
print(res)
```

^

SyntaxError: incomplete input

4. What is the difference between using a bare except block and specifying a specific exception type?

Solution:- Bare Except Block:- A bare except block catches the exception of all the types of exception in the try block. These sometimes creates difficulty to debug the code

Specific Exception:- Here we define the except block for particular type of exception in the try block. By using these debugging of the code is very easy to do

5. Can you have nested try-except blocks in Python? If yes, then give an example.

Solution:- Example of nested try-catch block

```
# Nested try-catch block
# If we will get the exception in the top most try block then the inner try block will not run it will directly the run exception of the upper most exception block
try :
    print("This is the outer try block")
    print(6/0)
    try :
        print("This is the inner try block ")
        print(8/0)
    except :
        print("This the inner except block")
except :
    print("This is the outer except block")

[48] ✓ 0.0s

...
This is the outer try block

This is the outer except block
```

```
# Nested try-catch block
# If we will get the exception in the top most try block then the inner try block will not run it will directly the run exception of the upper most exception block
try :
    print("This is the outer try block")
    # print(6/0)
    try :
        print("This is the inner try block ")
        print(8/0)
    except :
        print("This the inner except block")
except :
    print("This is the outer except block")

[50] ✓ 0.0s

...
This is the outer try block

This is the inner try block

This the inner except block
```

6. Can we use multiple exception blocks, if yes then give an example.

Solution:-

```
# Nested try-except block
try :
    num1 = int(input())
    num2 = int(input())
    res = (num1/num2)
    print(res)
except ZeroDivisionError :
    print("Denominator cannot be zero")
except ValueError:
    print("Please enter the numeric data")
```

[24] ✓ 2.6s

...

Denominator cannot be zero

```
# Nested try-except block
try :
    num1 = int(input())
    num2 = int(input())
    res = (num1/num2)
    print(res)
except ZeroDivisionError :
    print("Denominator cannot be zero")
except ValueError:
    print("Please enter the numeric data")
```

[26] ✓ 3.1s

...

Please enter the numeric data

7. Write the reason due to which following errors are raised:

a. EOFError

Solution:-

Occurs when Python has reached the end of user input without receiving any input

```
▶ # EOF Error
try :
    n = int(input())
    print(n*10)
except EOFError:
    print("EOF when reading a line")
```

b. FloatingPointError

Solution:-

```
▶ # Floating Point Error
# You may get sometime weired result when you are using the floating values
print(10.2-10.0)
# The wrong is coming due to floating point error
```

[52] ✓ 0.0s

... 0.19999999999999993

c. IndexError

Solution:-

These error comes when try to access the index which is not possile to access

d. MemoryError

Solution:-

the interpreter has run out of memory to allocate to your Python program.

e. OverflowError

Solution:-

An OverflowError exception is raised when an arithmetic operation exceeds the limits to be represented.

f. TabError

Solution:-

occurs when we mix tabs and spaces in the same code block.

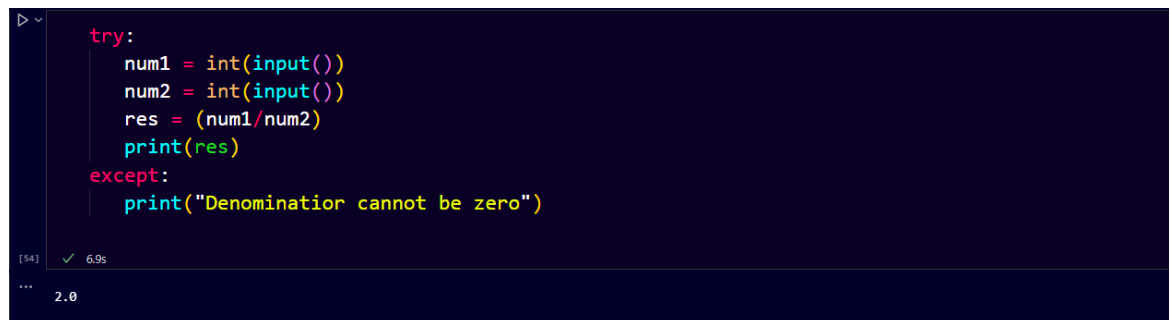
g. ValueError

Solution:-

If we are taking the input from the user as an integer format but we are giving the input in other data type like float or string

8. Write code for the following given scenario and add try-exception block to it.

a. Program to divide two numbers

Solution:-

```
try:
    num1 = int(input())
    num2 = int(input())
    res = (num1/num2)
    print(res)
except:
    print("Denominator cannot be zero")
```

[54] ✓ 6.9s

... 2.0

```
try:
    num1 = int(input())
    num2 = int(input())
    res = (num1/num2)
    print(res)
except:
    print("Denomination cannot be zero")
```

[55] ✓ 3.0s

... Denomination cannot be zero

b. Program to convert a string to an integer

Solution:-

```
# Programme to convert string to an integer
try :
    a = input()
    print(int(a))
except :
    print("You have entered the character")
```

[59] ✓ 3.9s

... 10

```
▶ # Programme to convert string to an integer
try :
    a = input()
    print(int(a))
except :
    print("You have entered the character")

[60] ✓ 3.0s
...
You have entered the character
```

c. Program to access an element in a list

Solution:-

```
▶ # Index Error
# These error comes when try to access the index which is not possible to access
try:
    index = int(input())
    lst=[1,4,5,6,8,9]
    print(lst[index])
except :
    print("You are entering the invalid index")

[62] ✓ 2.2s
...
5
```



```
▷ ▾  
# Index Error  
# These error comes when try to access the index which is not possible to access  
try:  
    index = int(input())  
    lst=[1,4,5,6,8,9]  
    print(lst[index])  
except :  
    print("You are entering the invalid index")  
[63] ✓ 3.3s  
...  
You are entering the invalid index
```

d. Program to handle a specific exception

Solution:-

```
▷ ▾  
# Progrmme to handle the Zero Division Error  
try :  
    num1 = int(input())  
    num2 = int(input())  
    res = (num1/num2)  
    print(res)  
except ZeroDivisionError :  
    print("Denominator cannot be zero")  
[72] ✓ 2.5s
```

```
▷ ▾ # Programme to handle the Zero Division Error
try :
    num1 = int(input())
    num2 = int(input())
    res = (num1/num2)
    print(res)
except ZeroDivisionError :
    print("Denominator cannot be zero")
except ValueError:
    print("Please enter the numeric data")

[65] ✓ 6.1s
... Denominator cannot be zero
```

e. Program to handle any exception

Solution:-

```
▷ ▾ try :
    a = int(input())
    print(a)
except :
    print("Please enter the integer value")

[66] ✓ 2.1s
... 10
```



```
try :  
    a = int(input())  
    print(a)  
except :  
    print("Please enter the integer value")
```

[67] ✓ 2.3s

... Please enter the integer value