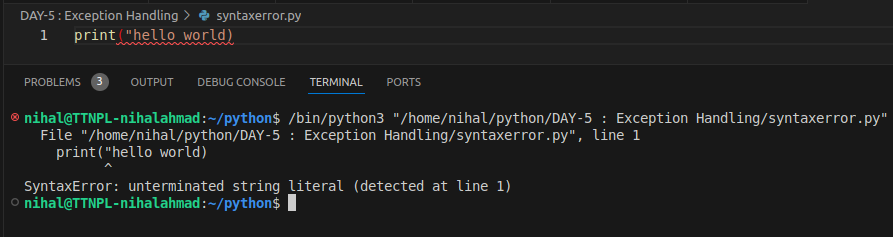
**EXCEPTION HANDLING**

1. **Define what are following exceptions, when to handle, and handle exceptions, below: SyntaxError, Exception, RuntimeError ValueError, TypeError, Warning**
2. **SyntaxError:**

The SyntaxErrors are raised when the python interpreter cant understand the code written by the programmer. This mainly occurs due to the following reasons:

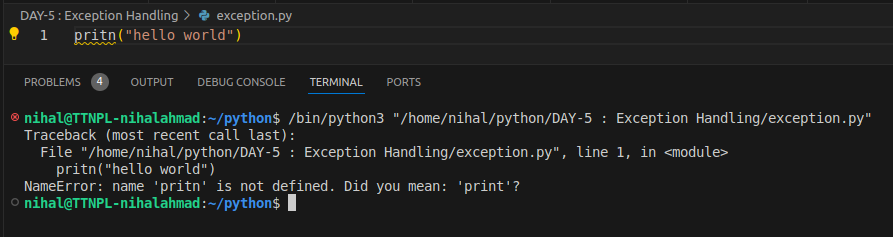
* Misspelled keywords
* Incorrect indentation
* Misplaced punctuations



1. **Exception:**

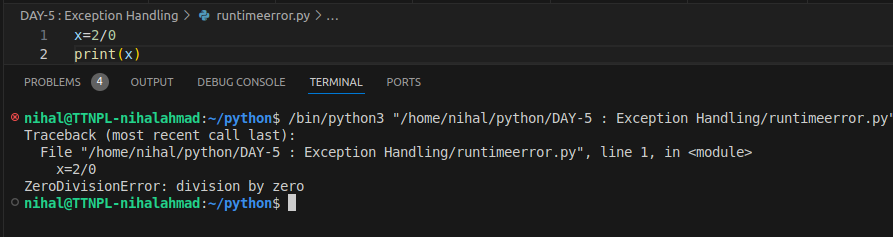
Sometimes the program still generate some errors even if its syntactically correct. These errors are mainly occurred during the execution of the program. The errors that are mainly encountered during the execution of the code is called exception. These can be occurred due to the following reasons:

* Incorrect operations
* Incorrect types of variable in an operation
* Calling a variable that is not defined in the program etc.



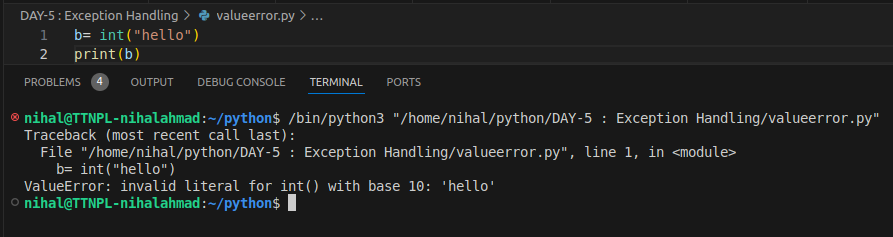
1. **RuntimeError:**

Runtime error is the error that is raised when the program is syntactically correct but the python interpreter still can't run the code due to some errors, these are called runtime errors.



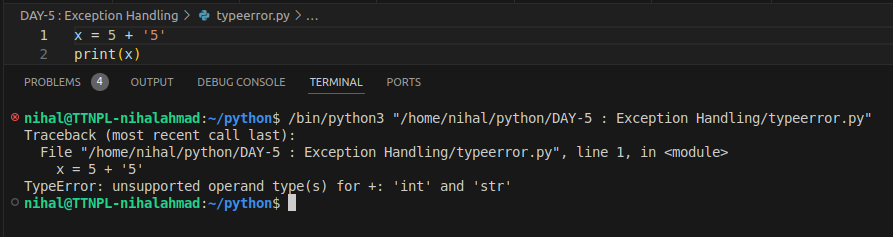
1. **ValueError:**

ValueError are the errors that are raised when the datatype of the variables is correct but the values provided in the variables are not correct.



1. **TypeError:**

TypeError is the error that is raised when the operations are performed on different types of operands.



1. **Warning:**

Warnings are provided to warn the developer of situations that aren’t necessarily exceptions. Unlike errors, the execution of a python program wont stop if there is a warning but will stop in case an exception is raised.

1. **How to define a custom exception? What are the occasions we should define a custom exception? Explain with code**

We can define a custom exception using the following syntax:

**class Class\_name(Exception):**

**#body of class**

We can create our own custom exception by inheriting the Exception class in our custom exception class.

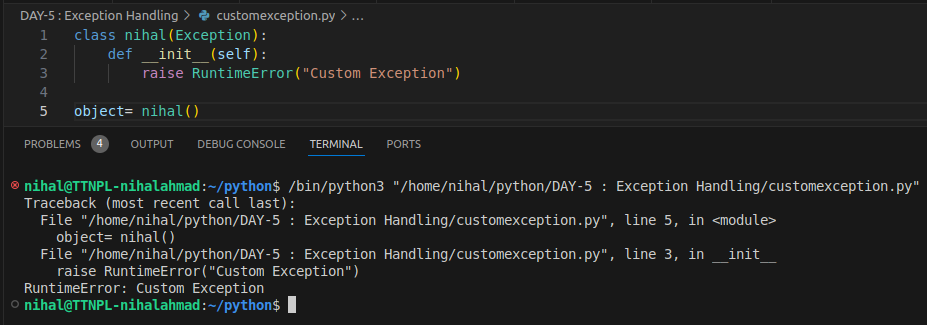
**When to define our own custom exceptions?**

We can define our own custom exceptions in the following cases:

* When we want to specifically define and raise an error in our program.
* When we want to make our code clear and more readable.
* It can also be helpful in debugging where we can program our exception to give more info about the errors so that we can have better understanding.

**Example:**

In the code block given below, I created a custom exception class “nihal” by inheriting the Exception class that raises a RuntimeError exception with a message “custom exception” whenever its object is created.

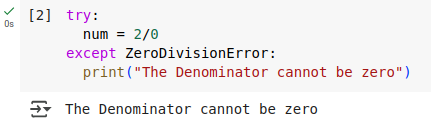
****

**EXTRAS:**

1. **Write a Python program to handle a ZeroDivisionError exception when dividing a number by zero.**

**Explanation:**

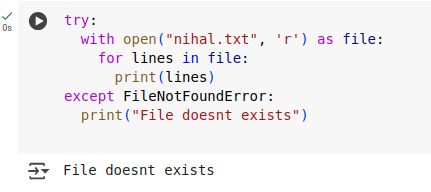
Here i declared a variable num in which we stored the value of 2/0 and since 2/0 is undefined then it will raise a ZeroDivisionError which is handled by the except block and the print statement gets executed.

****

1. **Write a Python program that opens a file and handles a FileNotFoundError exception if the file does not exist.**

**Explanation:**

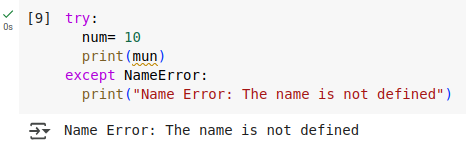
Here i tried opening a file nihal.txt which doesnt exist so it will raise a FileNotFoundError which is handled by the except block and the print statement gets executed.

****

1. **Write a Python program that raises a NameError exception and handle it.**

**Explanation:**

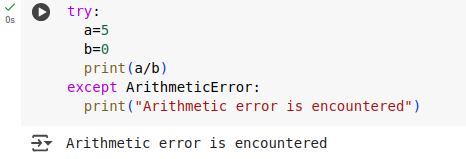
Here i declared a variable num in which we stored the value 10 but we printing a variable “mun” which is not declared in the code block and hence it will raise a NameError which is handled by the except block and the print statement gets executed.

****

1. **Write a Python program that executes division and handles an ArithmeticError exception if there is an arithmetic error.**

**Explanation:**

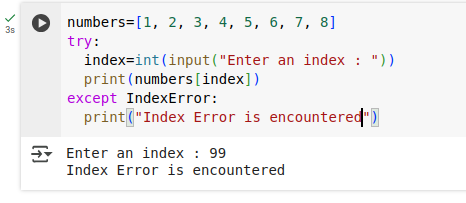
Here i declared two variable a and b and stored the value 5 and 0 respectively. I tried to print the value of a/b but it will raise an error as value of b is 0 and it will raise a ZeroDivisionError which is a part of ArithmeticError and hence it get handled by the except block and the print statement gets executed.

****

1. **Write a Python program that executes an operation on a list and handles an IndexError exception if the index is out of range.**

**Explanation:**

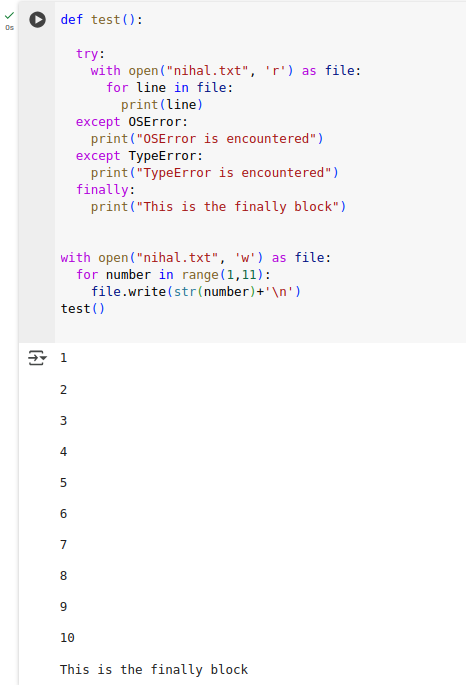
Here i declared a list of numbers with name numbers in which we stored 9 elements ie. 1, 2, 3, 4, 5, 6, 7, 8, 9 and i tried to take an input from user that is used to access the element on the specific index of the list but if we input a value big enough that it exceeds the number of elements in the list then it will raise an IndexError which is handled by the except block and the print statement gets executed.

****

1. **Write a python program to show the execution of finally statement.**

**Explanation:**

In this, we created a file “nihal.txt” and entered first 10 natural numbers in different lines. Then we created a function test() that read the file and handles OSError and TypeError and execute the respective print statements if the errors are encountered. A finally block is there to demonstrate its execution.

****