Exception Handling

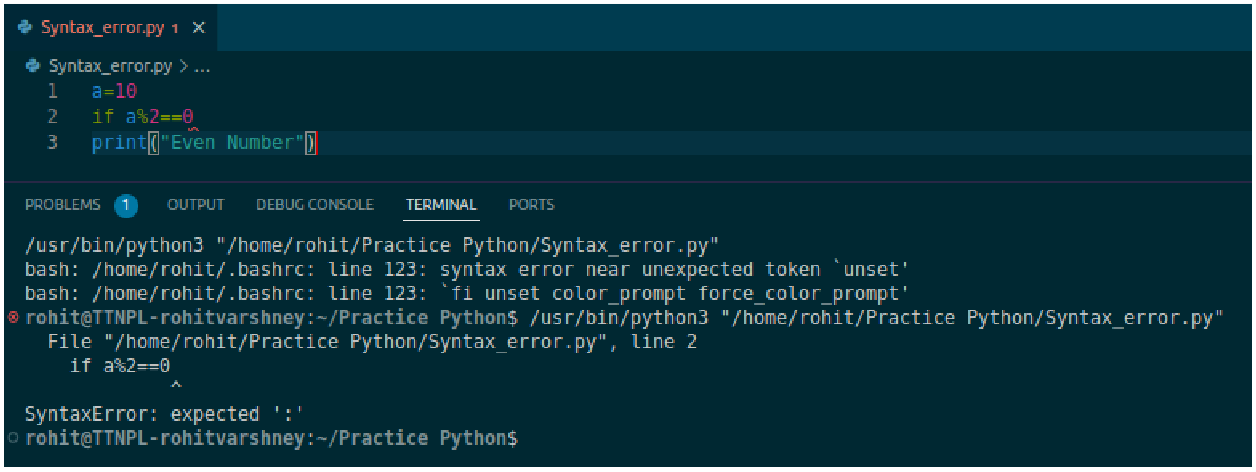
Q1) Define what are following exceptions, when to handle, and handle exceptions, below:
SyntaxError Exception RuntimeError ValueError TypeError Warning.

SyntaxError : -
Syntax errors are detected when we have not followed the rules of the particular programming language while writing a program. These errors are also known as parsing errors.
In python , syntax errors occur when python interpreter can't understand the structure of the statements in code .
The following reasons are the causes for Syntax Error are :

a) Incorrect indentation
b) Misspelled keywords
c) Incorrect punctuation

Example : a=20

```
if a%2==0                      # Syntax error : Missing Colon
    print "Even number"        # Syntax error : Missing Parenthesis
```



Syntax_error.py 1
Syntax_error.py > ...

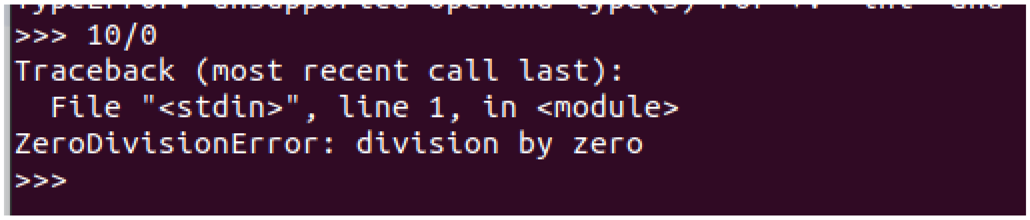
```
a=10
if a%2==0
print("Even Number")
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
/usr/bin/python3 "/home/rohit/Practice Python/Syntax_error.py"
bash: /home/rohit/.bashrc: line 123: syntax error near unexpected token `unset'
bash: /home/rohit/.bashrc: line 123: `fi unset color_prompt force_color_prompt'
rohit@TTNPL-rohitvarshney:~/Practice Python$ /usr/bin/python3 "/home/rohit/Practice Python/Syntax_error.py"
  File "/home/rohit/Practice Python/Syntax_error.py", line 2
    if a%2==0
             ^
SyntaxError: expected ':'
rohit@TTNPL-rohitvarshney:~/Practice Python$
```

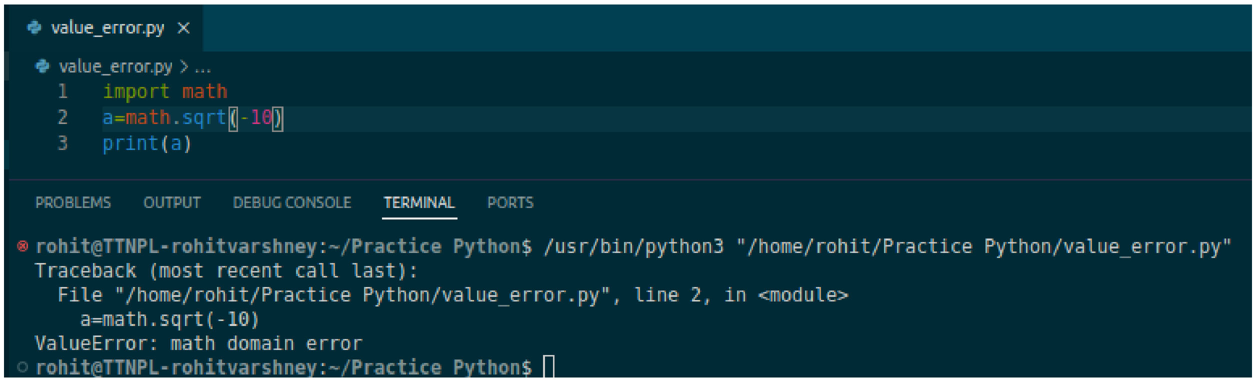
Exception : -
Exception refers to the the errors that will occur during the execution of the code even if the code is syntactically correct .

RuntimeError : - This occurs when the program is executing and encounters an unexpected condition that prevents it from continuing.



```
>>> 10/0
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero
>>>
```

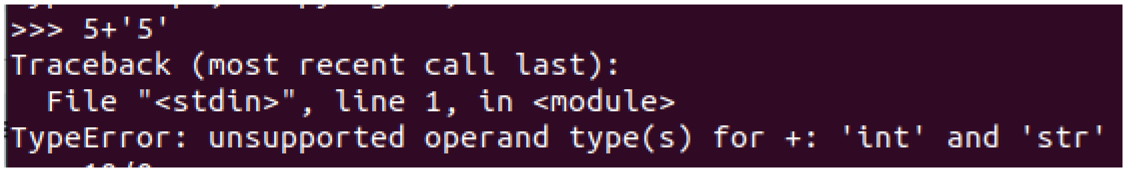
ValueError : - The ValueError occurs when an invalid value is assigned to a variable or passed to a function while calling it.



```
import math
a=math.sqrt(-10)
print(a)
```

```
rohit@TTNPL-rohitvarshney:~/Practice Python$ /usr/bin/python3 "/home/rohit/Practice Python/value_error.py"
Traceback (most recent call last):
  File "/home/rohit/Practice Python/value_error.py", line 2, in <module>
    a=math.sqrt(-10)
ValueError: math domain error
rohit@TTNPL-rohitvarshney:~/Practice Python$ 
```

TypeError : - It is occurred when an operator is supplied with a value of incorrect data type.



```
>>> 5+'5'
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Warning : - Warning is not an exception but it is used to inform or warn the programmer about the situations that may create some issues .

- **When to handle :** Warnings are handled when we want to inform other users about the potential issues , performance concerns etc .

Q2) How to define a custom exception? What are the occasions we should define a custom exception? Explain with code.

Custom Exception :

We can define custom exceptions by creating a new class. This exception class has to be derived from the built-in Exception class.

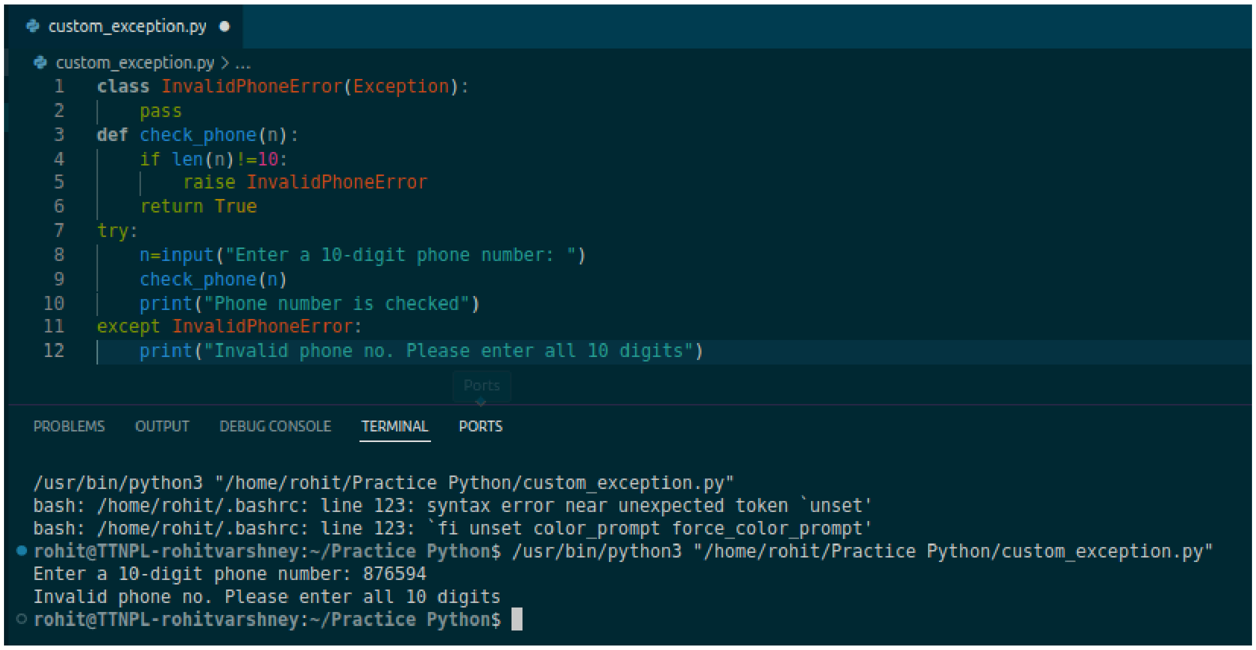
We can create custom exceptions to meet one’s requirements. These are called user-defined exceptions.

Syntax : class CustomError (Exception):

```
                    . . . .
        try :
              . . . .
       except CustomError :
              . . . .
```

Custom Exception can be defined in following cases :

a) It allows us to define our own error types to make our code more readable.
b) To handle specific cases that aren't covered by built-in exceptions.
c) Allows project specific exception handling.



```python
class InvalidPhoneError(Exception):
    pass
def check_phone(n):
    if len(n)!=10:
        raise InvalidPhoneError
    return True
try:
    n=input("Enter a 10-digit phone number: ")
    check_phone(n)
    print("Phone number is checked")
except InvalidPhoneError:
    print("Invalid phone no. Please enter all 10 digits")
```

```
/usr/bin/python3 "/home/rohit/Practice Python/custom_exception.py"
bash: /home/rohit/.bashrc: line 123: syntax error near unexpected token `unset'
bash: /home/rohit/.bashrc: line 123: `fi unset color_prompt force_color_prompt'
rohit@TTNPL-rohitvarshney:~/Practice Python$ /usr/bin/python3 "/home/rohit/Practice Python/custom_exception.py"
Enter a 10-digit phone number: 876594
Invalid phone no. Please enter all 10 digits
rohit@TTNPL-rohitvarshney:~/Practice Python$
```

CODE :

```
class InvalidPhoneError(Exception):
        pass
def check_phone(n):
        if len(n)!=10:
        raise InvalidPhoneError
        return True
try:
        n=input("Enter a 10-digit phone number: ")
        check_phone(n)
        print("Phone number is checked")
except InvalidPhoneError:
        print("Invalid phone no. Please enter all 10 digits")
```