# Python

# Exception Handling

# Python raise keyword

In this article, you will learn about the **raise** keyword of the **Python** programming language.

The **raise** is a predefined reserved word of Python. The exception handling technique in Python can be done through the keywords- try, except, finally, and raise. The **raise** statement manually creates an exception. We may use this to explicitly raise or throw an exception. The **raise** keyword is followed by determining the type of exception that you need to raise, which is indicated by either an object of the exception class or the name of the exception class. As we know, the try catch block is for handling exceptions, the raise keyword on the opposite is to generate an exception.

## Syntax of Python raise keyword

Here is the syntax of the **raise** keyword to explicitly raise an exception.

```
if test_condition:
   raise exception_type(argument, traceback)
```

Here, **exception_type** is a type of an exception to be raised explicitly. It could be characterized by either the name of an exception class or by an object of a special case class, that you are raising. So, we can define what kind of error to raise, and the text to print to the user. The **argument** is the string message related to the exception, which we are going to raise explicitly. This is an optional parameter, and if it is not defined, then its value is None. The **traceback** is an object. If the exception handler is not found, the program prints the **traceback** and exits. This is also an optional attribute.

## Example 1

The following example raises an error if the input matches with the specified string value.

```python
string = input("Enter a city name: ")

if string=="Delhi" or string=="Goa" or string=="Mumbai":
    raise Exception("This word is not allowed")
```

**Output of the above code-**

```
Enter a city name: Pune

Enter a city name: Delhi
Traceback (most recent call last):
  File "C:\python37\Scripts\projects\test.py", line 4, in
    raise Exception("This word is not allowed")
Exception: This word is not allowed
```

The following example handles the **ZeroDivisionError** type of exception.

```
x = int(input("Enter first number: "))
y = int(input("Enter second number: "))



try:
    # checking inputs for negative values
    if x < 0 or y < 0:
        raise ZeroDivisionError
    print(x/y)
except ZeroDivisionError:
    print("Please enter valid integer value")
```

**Output of the above code-**

```
Enter first number: 20
Enter second number: 5
4.0

Enter first number: 29
Enter second number: 0
Please enter valid integer value
```