

# Python Error and its Handling

Exception/ Error/ Bug is unusual condition that occur in python program. The error in program can give you un-expected output or may be cause of abnormal termination of program execution.

## Types of Error in Python

1. Syntax Error
2. Runtime Error
3. Logical Error
4. Exception

### 1. Syntax Error

These type of error occurred due to violation of Python syntax in program.

Example-1:

```
print("Hello , This is Python)
```

Error: The string given in print() is not closed with " mark. It is syntax error.

Example-2:

```
A=10
```

```
B=20
```

```
A+B=Sum
```

Error: A+B=Sum is violation of python assignment syntax. It should be Sum=A+B

## 2. Runtime Error

Runtime errors generate at the time of running of Python Program.

Example

- Variable not declared.

```
Sum=A+10    # Here A is not defined
```

- Illegal value assigned to variable

```
Name= "KV" + 2 # integer cannot add in String
```

## 3. Logical Error

Logical Errors are related to wrong logic applied in Python statement. These errors are very difficult to identify in program because these are not syntax error. Logical error produce wrong output but will not show any error.

Example:

- Find the sum of 2 numbers

```
A=10
```

```
B=20
```

```
Sum=A * B    # Logical Error
```

```
# It will give the multiplication of 2 numbers in place  
of Addition.
```

## 5. Exception

Exceptions are errors which occur during the execution of program. The exceptions are triggered automatically and cause the abnormal termination of program.

Example:

```
A=int(input("Enter number-1: "))
```

```
B=int(input("Enter number-2: "))
```

```
R=A/B
```

Exception: if the value of B became ZERO then the infinite value type exception will create and the program will terminate abnormally.

## 6. Exception Handling:

The code that prevents the abnormal termination of program while an exception occurs. This code is known as exceptional handling.

Exceptions can be handle with following method.

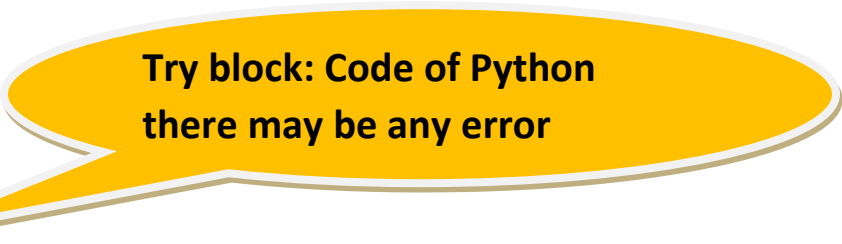
```
try:
```

```
    A=10
```


```
    B=20
```

```
    R=A/B
```

```
    print(R)
```



**Try block: Code of Python  
there may be any error**



**except block: Execute this code  
when there is an exception**

```
except:
```

```
    print(" Divide by ZERO, not possible")
```

Table 7.1: Common Python Exceptions

Exception	Description
<b>NameError</b>	Raised when an identifier is not found in the local or global namespace.
<b>TypeError</b>	Raised when an operation or function is attempted that is invalid for the specified data type.
<b>ValueError</b>	Raised when a built-in operation or function receives an argument that has the right type but an inappropriate value.
<b>ZeroDivisionError</b>	Raised when division or modulo by zero takes place for all numeric types.
<b>AttributeError</b>	Raised when an object does not find an attribute.
<b>KeyError</b>	Raised when a mapping (dictionary) key is not found in the set of existing keys.
<b>IndentationError</b>	Raised due to incorrect indentation.
<b>IOError</b>	Raised if the file requested cannot be opened, or failure of I/O operation.
<b>IndexError</b>	Raised when an index is not found in a sequence, <i>i.e.</i> , out of range or out of bound.
<b>ImportError</b>	Raised if Python cannot find the module requested for in a program.
<b>EOFError</b>	Raised when one of the file methods ( <i>i.e.</i> , <code>read()</code> , <code>readline()</code> or <code>readlines()</code> ) tries to read beyond the file.
<b>SyntaxError</b>	Raised when there is an error in Python syntax.
<b>RuntimeError</b>	Raised when an error does not fall under any specific exception category defined above.

## Try and except clause

### try:

In Python Exception handling, the actual program should be written under try block. If any exception / error raised in program, then it will be smartly handled at the runtime of program.

### except:

The except block is the exception/ error controlling block. When any exception raised in try block then the code written under except

block will be execute and program terminate properly.

Example:

try:

```
A=10
```

```
B=20
```

```
print("Sum of A & B: ", A+B)
```

```
print("Subtraction of A & B: ", A-B)
```

```
print("Multiplication of A & B: ", A*B)
```

```
print("Division of A & B: ", A/B)
```

```
print("Power of A & B: ", A**B)
```

```
print("Floor Division of A & B: ", A//B)
```

except:

```
print("Something wrong in program")
```

```
print("Kindly check input values")
```

The above program will execute properly and never terminate abnormally.

**\*\*Finish\*\***