Exception

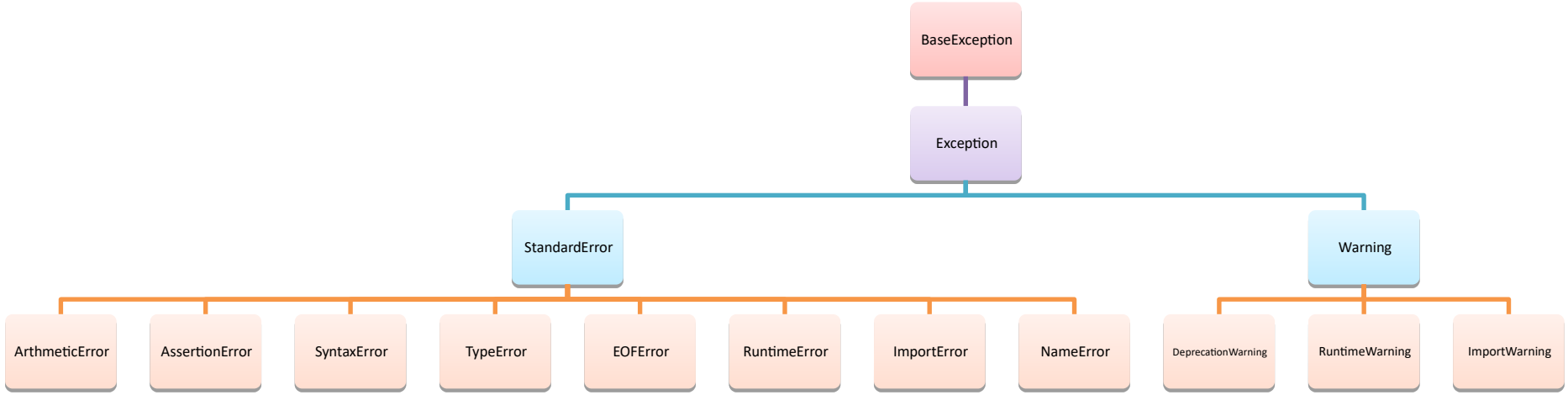
An exception is a runtime error which can be handled by the programmer.

All exceptions are represented as classes in Python.

Type of Exception:-

- Built-in Exception – Exceptions which are already available in Python Language. The base class for all built-in exceptions is BaseException class.
- User Defined Exception – A programmer can create his own exceptions, called user-defined exceptions.

All exceptions are represented as classes in Python.



Need of Exception Handling

- When an exception occurs, the program terminates suddenly.
- Suddenly termination of program may corrupt the program.
- Exception may cause data loss from the database or a file.

Exception Handling

Try – The try block contains code which may cause exceptions.

Syntax-

```
try:
    statements
```

Except – The except block is used to catch an exception that is raised in the try block. There can be multiple except block for try block.

Syntax-

```
except ExceptionName:
      statements
```

Exception Handling

Else – This block will get executed when no exception is raised. Else block is executed after try block.

Syntax-

```
else:
    statements
```

Finally – This block will get executed irrespective of whether there is an exception or not.

Syntax-

```
finally:
    statements
```

- We can write several except blocks for a single try block.
- We can write multiple except blocks to handle multiple exceptions.
- We can write try block without any except blocks.
- We can not write except block without a try block.
- Finally block is always executed irrespective of whether there is an exception or not.
- Else block is optional.
- Finally block is optional.

```
try:
    Statement
except ExceptionClassName:
   Statement
else:
  Statement
finally:
   Statement
```

```
try:
    Statement
except ExceptionClassName:
   Statement
```

```
try:
    Statement
except ExceptionClassName1:
   Statement
except ExceptionClassName2:
   Statement
finally:
   Statement
```

```
try:
    Statement
```

```
except ExceptionClassName:
   Statement
```

Except

- With the Exception Class Name

```
except ExceptionClassName:
    Statement
```

- Exception as an object

```
except ExceptionClassName as obj:
    Statement
```

- Multiple Exception within tuple

```
except (ExceptionClassName1, ExceptionClassName2, ExceptionClassName3, …… ):
    Statement
```

- Catch any Type of Exception

```
except:
    Statement
```

Assert Statement

The assert Statement is useful to ensure that a given condition is True. If it is not true, it raises AssertionError.

Syntax:- assert condition, error_message

- If the condition is False then the exception by the name AssertionError is raised along with the message.
- If message is not given and the condition is False then also AssertionError is raised without message.

User Defined Exception

A programmer can create his own exceptions, called user-defined exceptions or Custom Exception.

- Creating Exception Class using Exception Class as a Base Class
- Raising Exception
- Handling Exception

Creating Exception

We can create our own exception by creating a sub class to built-in Exception class.

```
class MyException(Exception):
        pass


class MyException(Exception):
        def __init__(self, arg):
       self.msg = arg
```

Raising Exception

raise statement is used to raise the user defined exception.

raise MyException(‘message’)

Handling Exception

Using try and except block Programmer can handle exceptions.

```
try:
   statement
except MyException as mye:
   statement
```

Error vs Exception

- An exception is an error that can be handled by a programmer.
- An exception which are not handled by programmer, becomes an error.
- All exceptions occur only at runtime.
- Error may occur at compile time or runtime.

Error vs Warning

It is compulsory to handle all error otherwise program will not execute, while warning represents a caution and even though it is not handled, the program will execute.

Errors are derived as sub class of *StandardError*, while warning derived as sub class from *Warning* class.