Experiment No. 8

Aim: To implement programs based on Exception Handling.

Questions:

1. Note down various pre-defined exceptions in Python.

```
# ArithmeticError:- Raised when an error occurs in numeric calculations
# AssertionError:- Raised when an assert statement fails
# AttributeError:- Raised when attribute reference or assignment fails
# Exception:- Base class for all exceptions
# EOFError:- Raised when the input() method hits an "end of file" condition (EOF)
# FloatingPointError:- Raised when a floating point calculation fails
# GeneratorExit:- Raised when a generator is closed (with the close() method)
# ImportError:- Raised when an imported module does not exist
# IndentationError:- Raised when indendation is not correct
# IndexError:- Raised when an index of a sequence does not exist
# KeyError:- Raised when a key does not exist in a dictionary
# KeyboardInterrupt:- Raised when the user presses Ctrl+c, Ctrl+z or Delete
```

2. Write a python program to demonstrate try and except block. (with specific exception, without specific exception and with multiple exceptions)

```
In [26]: def func(a):
             if a < 4:
                 b = a / (a - 3)
             print("The value of b:", b)
             print(c)

         try:
             func(4)
             func("abc")
         except UnboundLocalError:
             print("UnboundLocalError Handled")
         except (TypeError, NameError) as e:
             print(e)
         except:
             print("")
```

```
UnboundLocalError Handled
```

3. Write a python program to demonstrate try, except, else and finally block.\

```
In [34]: def func(a):
             print(a + 10)

         try:
             a = int(input("Enter a number: "))
             func(a)
         except ValueError:
             print("Please Enter a integer")
         else:
             print(":::")
         finally:
             print("Hello!!")
```

```
Enter a number: abc
Please Enter a integer
Hello!!
```

4. Write a python program to demonstrate user defined exceptions.

```
In [35]: class AgeError(Exception):
             pass
         age = int(input("Enter your age: "))
         if age < 18:
             raise AgeError("Age Less Than 18")
```

```
Enter your age: 17

---------------------------------------------------------------------------
AgeError                                  Traceback (most recent call last)
<ipython-input-35-42f9e7285454> in <module>
      3 age = int(input("Enter your age: "))
      4 if age < 18:
----> 5     raise AgeError("Age Less Than 18")

AgeError: Age Less Than 18
```

5. Write a python program to demonstrate assertions.

```
In [38]: try:
             age = int(input("Enter your Age: "))
             assert age >= 18, "Invalid Age"
         except AssertionError as e:
             print("Entered age is below 18")
             print(e)
```

```
Enter your Age: 17
Entered age is below 18
Invalid Age
```