```
User-Defined Exceptions:
------------------------
        we can define a exception according to our business requirements,that type
of exceptions are called User-defined exceptions.

how to implement the user-defined Exceptions:
---------------------------------------------
step1:
-----
to create a user-defined Exception class.

        class classname(Exception):
                ---------
                ---------
                ---------

ex:
---
        class Test(Exception):
                ------
                ------
                ------

        here Test is a user-defined Exception class

step2:
-----
to raise the Exceptions

        we can raise the exceptions manually by using 'raise' keyword.

        raise ExceptionClassname

ex:
---
        raise Test

step3:
-----
to handle the user-defined exceptions

        we can handle the user-defined exception's by using try and except blocks.

ex:
---
class ValueTooSmall(Exception):
    """Value Too Small Userdefined Exception Class"""

class ValueTooLarge(Exception):
    """Value Too Large Userdefined Exception Class"""
```

```
import random
x=random.randint(0,9)
while True:
    y=int(input("enter y value: "))
    try:
        if y>x:
            raise ValueTooLarge
        elif y<x:
            raise ValueTooSmall
        else:
            break
    except ValueTooLarge:
        print("value too large,Try again!")
    except ValueTooSmall:
        print("value too small,Try again!")
print("Bye")
```

output:

```
enter y value: 7
value too large,Try again!
enter y value: 3
value too small,Try again!
enter y value: 5
value too large,Try again!
enter y value: 4
Bye
```

Assertion's:
----

assertion is a boolean expression,whenever assertion to return the True do-nothing in our program,our program contineously executed otherwise to-stop the execution to raise the AssertionError.

we can implement the assertions in python by using assert keyword.

syntax

```
        assert condition

                (or)

        assert condition,"error message"
```

ex1:

```
obj=eval(input("enter your iterable object: "))
def my_avg(x):
    assert len(x)!=0
    print(sum(x)/len(x))
my_avg(obj)

output: without exception
------
enter your iterable object: [3,2,7,1]
3.25

output2: with exception
--------
enter your iterable object: []

AssertionError

ex2:
----
obj=eval(input("enter your iterable object: "))
def my_avg(x):
    assert len(x)!=0,"length is zero"
    print(sum(x)/len(x))
my_avg(obj)

output1: without any exception
--------
enter your iterable object: [3,2,7,1]
3.25

output2: with exception
-------
enter your iterable object: []
AssertionError: length is zero

ex3:
----
class strclass(Exception):
    """string exception class"""
class noniterable(Exception):
    """non-iterable exception class"""
obj=eval(input("enter your iterable object: "))
def my_avg(x):
    try:
        if type(x)==str:
            raise strclass
        elif type(x) in [int,float,complex]:
            raise noniterable
        else:
            assert len(x)!=0
    except strclass:
        print("we can't perform average operation on strings")
    except noniterable:
        print("please enter iterable objects only")
    except AssertionError:
        print("length should not be zero")
    else:
        print(sum(x)/len(x))
my_avg(obj)

output1:
--------
enter your iterable object: "siva"
we can't perform average operation on strings

output2:
--------
enter your iterable object: 4
please enter iterable objects only

output3:
--------
enter your iterable object: 2.3
please enter iterable objects only

output4:
--------
enter your iterable object: [5,3,7,2]
4.25

output5:
--------
enter your iterable object: []
length should not be zero
```