WORKING WITH PYTHON LIST DATA STRUCTURE
A list is a container which holds comma-separated values (items or elements) between square brackets[ ] where items or elements need not all have the same type. It can have any number of items and they may be of different types (integer, float, string etc.).

List has the following five characteristics:
1. MUTABLE
2. Linear Data Structure
3. Mixed Type Elements
4. Variable Length
5. Zero Based Indexing

NOTE: Traditional arrays can not be created in Python.

Types of lists :
Empty List: A list without any element is called an empty list

Example:
```
PyList = []
print(PyList)#[]
```

Number List:
1 Integers List:
A list with only numbers is called an integer list

Example:
```
PyList=[1,2,3,4,5]
print(type(PyList))#<class 'list'>
print(PyList)#[1,2,3,4,5]
```

Float List:
A list with only decimal numbers is called float list

Example:
```
PyList=[1.1,2.88,3.33,4.1,5.0]
print(type(PyList))#<class 'list'>
print(PyList)#[1.1,2.88,3.33,4.1,5.0]
```

String List:
A list with only Strings & Chars is called a string list

Example:
```
PyList = ["Sara", "David", "Raju", "Sandy"]
print(PyList)#['Sara', 'David', 'Raju', 'Sandy']
print(type(PyList))#<class 'list'>
```

Mixed List:
A list with different datatypes is called Mixed list.

Example:
```
PyList=["Sara", 1, 2.03,'A']
print(type(PyList))#<class 'list'>
print(PyList)#['Sara', 1, 2.03,'A']
```

Nested List:

A list with in another list is called Nested List.

Example:
```
PyList=["Mouse", [8, 4, 6], ['a']]
print(PyList)#['Mouse', [8, 4, 6], ['a']]
print(type(PyList))#<class 'list'>
```

Example:Dynamic List
```
PyList=eval(input("Enter List: "))
print(type(PyList))#<class 'list'>
print(PyList)
```

Basic List Operations
We can perform the following basic operations on list data structure.
1.Concatenation
2. Repetition
3. Membership
4. Iteration
5. Length

Len Example:(Length)
```
PyDataSet=[1,2,4,4]
print(len(PyDataSet))#4
```

Example2: Concatenation
```
PyDataSet1=[1,2,4,4]
PyDataSet2=['a','b','c','d']
PyDataSet3=PyDataSet1+PyDataSet2
print(PyDataSet3)#[1,2,4,4,'a','b','c','d']
```

Example: Repetition
```
PyDataSet=[1,2]
print(PyDataSet*4)#[1,2,1,2,1,2,1,2]
```

Example : Membership
```
PyDataSet=[1,2,3]
print(1 in PyDataSet)#True
```

Example: Iteration
```
PyList=['Raju','Smith','Sara','Scott']
for friend in PyList:
    print("Say Hey : ",friend)
```

How to Access elements from a list?
There are various ways in which we can access the elements of a list.

List Index
We can use the index operator [] to access an item in a list. Index starts from 0. So, a list having 5 elements will have index from 0 to 4. The index must be an integer. Nested list are accessed using nested indexing.

NOTE:
We can't use float or other types as index, this will result into TypeError.

Example:

```
PyList=["Big Data", "Hadoop", "Spark", "IoT"]
Item              Big Data       Hadoop  Spark   IoT
Index (from left)        0        1       2       3
Index (from right)      -4       -3      -2      -1
```

Example:

```
PyList = ['P','Y','T','H','O','N']
print(PyList[0])
print(PyList[4])
print(PyList[-1])
print(PyList[-4])
```

Negative indexing

Python allows negative indexing for its sequences. The index of -1 refers to the last item, -2 to the second last item and so on.

Example:

```
PyList=["Big Data", "Hadoop", "Spark", "IoT"]
print(PyList[0])
print(PyList[0],PyList[3])
print(PyList[-1])
print(PyList[4])
```