CLASS XI
COMPUTER SCIENCE
STUDY MATERIAL
PYTHON – LIST MANIPULATION

Introduction

List is a sequence of items separated by commas and items are enclosed in square brackets [ ].

- Is a container that are used to store a list of values of any type
- List is mutable type i.e. we can change value in place without creating a fresh list
- List is a type of sequence like tuples and string but in differs them in the way that lists are mutable but string and tuples are immutable
- List can have elements of different data types such as integer, float, string, tuple or even another list.

```
#list1 is the list of six even numbers
>>> list1 = [2,4,6,8,10,12]
>>> print(list1)
[2, 4, 6, 8, 10, 12]
#To create a list
>>> list1 = [5, 3.4, "New Delhi", "20C", 45] #print the elements of the list list1
>>> list1
[5, 3.4, 'New Delhi', '20C', 45]
```

Creating List

To create a list the following syntax we need to follow:
ListName = [ ] Or
ListName = [value1, value2,…..]
For example
Family = [“father”,”mother”,”brother”,”sister”]
Student = [1,”Aman”,”XI”,3150]
The above construct is known as list display construct
Consider more examples
EMPTY LIST
L = [ ] Or
L = list()

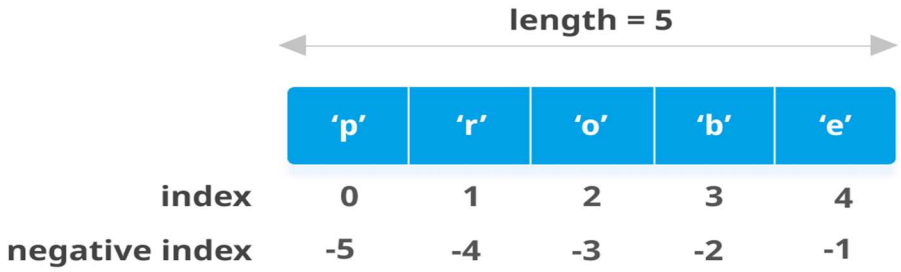
Type of Lists

```
#list2 is the list of vowels (characters)
>>> list2 = ['a','e','i','o','u']
>>> print(list2)
['a', 'e', 'i', 'o', 'u']

#list3 is the list of mixed data types
>>> list3 = [100,23.5,'Hello']
>>> print(list3)
[100, 23.5, 'Hello']

#list1 is the list of integer data type
```

```
>>> list1 = [100,2,23,5]
>>> print(list1)
[100, 2,23,5]

#list4 is the list of lists called nested #list
>>> list4 =[['Physics',101],['Chemistry',202], ['Mathematics',303]]
>>> print(list4)
[['Physics', 101], ['Chemistry', 202],
['Mathematics', 303]]
```

ACCESSING ELEMENTS IN A LIST

Each element in list is accessed using value called **index**.
The first index value is 0, the second index is 1 and so on. Elements in the list are assigned index values in increasing order sterling from 0. This is called as **indexing.**
To access an element, use square brackets with the index [] value of that element.
We may also use negative index value to access elements starting from the last (backward indexing) element in the list, having index value -1.
Forward indexing starts from the left side with 0 as the first index.




```
#initialing a list named list1
>>> list1 = [2,4,6,8,10,12]
>>> list1[0] #returns first element of list1
2
>>> list1[3] #returns fourth element of list1
8
#Out of range index value for the list returns error
>>> list1[15]
IndexError: list index out of range
#an expression resulting in an integer index

>>> list1[1+4]
12
>>> list1[-1] #return first element from right
12
#length of the list1 is assigned to n
```

```
>>> n = len(list1)
>>> print(n)
6

#Get the last element of the list1
>>> list1[n-1]
12
#Get the first element of list1
>>> list1[-n]
2
```

CREATING LIST FROM EXISTING SEQUENCE

We can also create list of single characters or single digits through keyboard input:

For example:

```
>>> list1 = list (input (“Enter list elements”))
>>> list1
```

From the above code whatever values we will enter will be of STRING type. To store list of integers through input in python we can use eval () to convert the list items to integers

```
>>> list1 = eval (input (“enter list to be entered”))
>>> print (“list is “, list1)
```

eval ()

```
>>> list1 = eval(input(“Enter values :”))
>>> enter values: [10,20,30]
>>> list1  [10,20,30]
```

Lists are Mutable

In Python, lists are mutable. It means that the contents of the list can be changed after it has been created.

```
#List list1 of colors
>>> list1 = ['Red','Green','Blue','Orange’]
#change/override the fourth element of list1
>>> list1[3] = 'Black’
>>> list1 #print the modified list list1
['Red', 'Green', 'Blue', 'Black']
```

LIST OPERATIONS

CONCATENATION (JOINING STRINGS)

Python allows us to join two or more lists using concatenation operator using symbol +.

```
#list1 is list of first five odd integers
>>> list1 = [1,3,5,7,9]
#list2 is list of first five even integers
>>> list2 = [2,4,6,8,10]
#Get elements of list1 followed by list2
>>> list1 + list2
[1, 3, 5, 7, 9, 2, 4, 6, 8, 10]
>>> list3 = ['Red','Green','Blue']
>>> list4 = ['Cyan', 'Magenta', 'Yellow' ,'Black']
>>> list3 + list4
['Red','Green','Blue','Cyan','Magenta', 'Yellow','Black']
```

Note that, there is no change in original lists i.e., list1, list2, list3, list4 remain the same after concatenation operation. If we want to use the result of two concatenated lists, we should use an assignment operator.

For example,

```
#Join list 2 at the end of list
>>> new List = list 1 + list 2
>> new list

[1, 3, 5, 7, 9, 2, 4, 6, 8, 10]
```

The concatenation operator '+' requires that the operands should be of list type only. If we try to concatenate a list with elements of some other data type, Type Error occurs.

```
>>> list1 = [1,2,3]

>>> str1 = "abc"
>>> list1 + str1
Type Error: can only concatenate list (not "str") to list
```

REPETITION

Python allows us to replicate the contents of a list using repetition operator depicted by symbol *.

```
>>> list1 = ['Hello']
#elements of list1 repeated 4 times
>>> list1 * 4
['Hello', 'Hello', 'Hello', 'Hello']
```

MEMBERSHIP

The membership operator in checks if the element is present in the list and returns True, else returns False.

```
>>> list1 = ['Red','Green','Blue']
>>> 'Green' in list1
True
>>> 'Cyan' in list1
False
```

The Operator not in transpose returns True if the element is not present in the list, else it returns False.

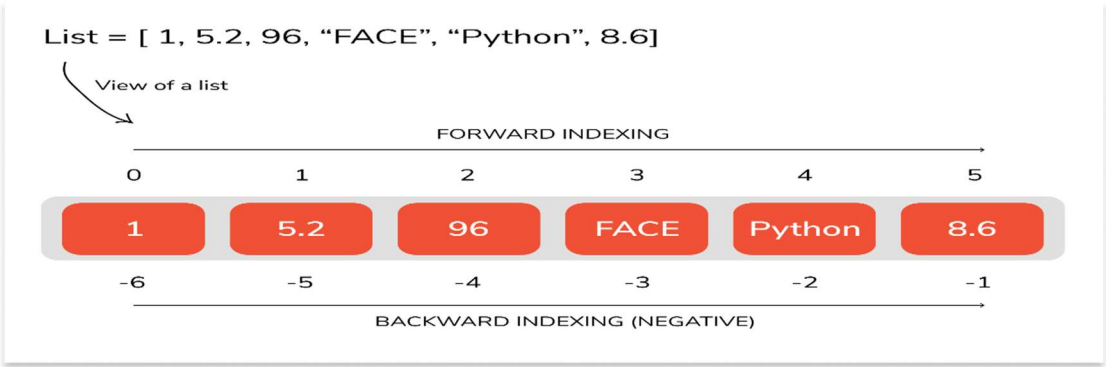
```
>>> list1 = ['Red','Green','Blue']
>>> 'Cyan' not in list1
True
>>> 'Green' not in list1
False
```

SLICING

Slicing operations allow us to create new list by taking out elements from an existing list.

```
>>> list1 =['Red','Green','Blue','Cyan','Magenta','Yellow','Black']
#subject from indexes 2 to 5 of list 1
>>> list1[2:6]
['Blue', 'Cyan', 'Magenta', 'Yellow']

#list1 is truncated to the end of the list

>>> list1[2:20]                    #second index is out of range
['Blue', 'Cyan', 'Magenta', 'Yellow', 'Black']
```

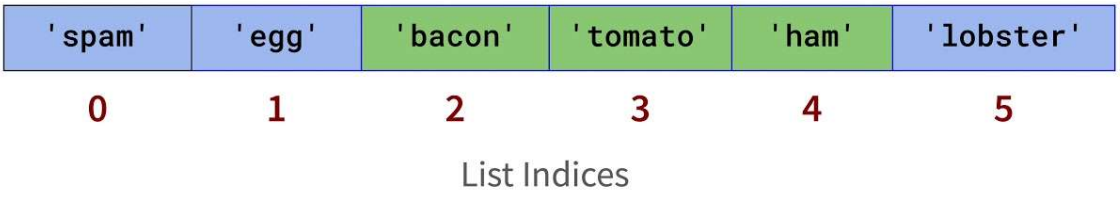
```
>>> list1[7:2]          #first index > second index
[ ]                      #results in an empty list
                                 #return sublist from index 0 to 4
>>> list1[:5]                    #first index missing
['Red','Green','Blue','Cyan','Magenta']
#slicing with a given step size

>>> list1[0:6:2]
['Red','Blue','Magenta']

#negative indexes
#elements at index -6, -5, -4, -3 are sliced
>>> list1[-6: -2]
['Green','Blue','Cyan','Magenta']
#both first and last index missing
>>> list1[::2] #step size 2 on entire list
['Red','Blue','Magenta','Black']
#Access list in the reverse order using negative step size
>>> list1[::-1]
['Black','Yellow','Magenta','Cyan','Blue', 'Green','Red']
```



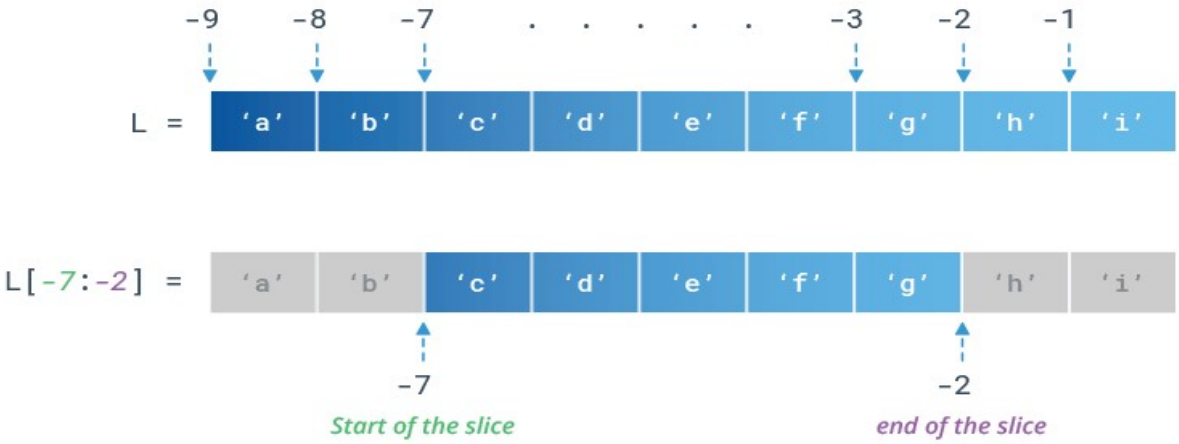
LISTS - INDEXING AND SLICING

List Slicing

```
a[2:5]
```



List Indices



LIST OPERATIONS

TRAVERSING A LIST

We can access each element of the list or traverse a list using a for loop or a while loop.

1. **List traversal using for in loop:**

```
>>> list1 = ['Red','Green','Blue','Yellow', 'Black']
>>> for item in list1:
        print(item)
```

Output:

Red

Green

Blue

Yellow

Black

2. **Using for i in range() loop**

Another way of accessing the elements of the list is using **range()** and len() functions:

```
>>> for i in range(len(list1)):
        print(list1[i])
```

Output:

Red

Green

Blue

Yellow

Black

```
val = [10,20,30,40,1,2,3,100,200,10,20,30,11,12,15,17,19,90,77,35]

slice1 = val[5:15:2]
slice2 = val[::4]
sum = 0

for x in slice1:
    print(x,"==“, end='')
    sum+=x
print("Sum of slice1 elements are “, sum)


sum=0

for x in slice2:
    print (x,"== “, end='')
    sum+=x
avg = sum / len(slice2)

print("Average of slice 2 = “, avg)
```

USING SLICING FOR LIST MODIFICATION

```
items=["One","Two","Three","Four”] items[0:2] =[1,2]
for i in items:
        print (i, end=' ')
   items [0:3] ="Fantastic"
   print()
   for i in items:
        print(i,end=' ')
```

LIST METHODS AND BUILT-IN FUNCTIONS

The data type list has several built-in methods that are useful in programming.

len()

Returns the length of the list passed as the argument >>> list1 = [10,20,30,40,50]

```
>>> len(list1)
5
```

list()

Creates an empty list if no argument is passed

```
>>> list1 = list()
>>> list1[ ]
```

Creates a list if a sequence is passed as an argument

```
>>> str1= 'aeiou'
>>> list1 = list(str1)
>>> list1
['a', 'e', 'i', 'o', 'u']
```

append ()

Appends a single element passed as an argument at the end of the list

A list can also be appended as an element to an existing list

```
>>> list1 = [10,20,30,40]
>>> list1.append(50)
>>> list1
[10, 20, 30, 40, 50]
```

```
>>> list1 = [10,20,30,40]
>>> list1.append([50,60])
>>> list1
[10, 20, 30, 40, [50, 60]]
```

extend()

Appends each element of the list passed as argument at the end of the given list.

```
>>> list1 = [10,20,30]
>>> list2 = [40,50]
>>> list1.extend(list2)
>>> list1
[10, 20, 30, 40, 50]
```

insert()

Inserts an element at a particular index in the list

```
>>> list1 = [10,20,30,40,50]
#inserts element 25 at index value 2
>>> list1.insert(2,25)
>>> list1
[10, 20, 25, 30, 40, 50]
>>> list1.insert(0,100)
>>> list1
[100, 10, 20, 25, 30, 40, 50]
```

count ()

Returns the number of times a given element appears in the list

```
>>> list1 = [10,20,30,10,40,10]
>>> list1.count(10)
3
>>> list1.count(90)
0
```

index ()

Returns index of the first occurrence of the element in the list. If the element is not present, Value Error is generated

>>> list1 = [10,20,30,20,40,10]

>>> list1.index(20)

1

>>> list1.index(90)

Value Error: 90 is not in list

remove ()

Removes the given element from the list. If the element is present multiple times, only the first occurrence is removed. If the element is not present, then Value Error is generated

>>> list1 = [10,20,30,40,50,30]

>>> list1.remove(30)

>>> list1

[10, 20, 40, 50, 30]

>>> list1.remove(90)

Value Error: list. Remove(x):x not in list

pop()

Returns the element whose index is passed as argument to this function and also removes it from the list. If no argument is given, then it returns and removes the last element of the list

>>> list1 = [10,20,30,40,50,60]

>>> list1.pop(3)

40

>>> list1

[10, 20, 30, 50, 60]

>>> list1 = [10,20,30,40,50,60]

>>> list1.pop ()

60

>>> list1

[10, 20, 30, 40, 50]

reverse ()

Reverses the order of elements in the given list

>>> list1 = [34,66,12,89,28,99]

>>> list1.reverse()

>>> list1

[99, 28, 89, 12, 66, 34]

>>> list1 = ['Tiger' ,'Zebra' , 'Lion' , 'Cat' ,'Elephant' ,'Dog']

>>> list1.reverse()

>>> list1

['Dog', 'Elephant', 'Cat', 'Lion', 'Zebra', 'Tiger']

sort ()

Sorts the elements of the given list **in place** (means changes are made to the original list).

>>>list1 = ['Tiger','Zebra','Lion', 'Cat', 'Elephant' ,'Dog']

>>> list1.sort()

>>> list1

['Cat', 'Dog', 'Elephant', 'Lion', 'Tiger', 'Zebra']

>>> list1 = [34,66,12,89,28,99]

>>> list1.sort(reverse = True)

>>>list1

[99,89,66,34,28,12]

sorted()

It takes a list as parameter and creates a **new list** consisting of the same elements but arranged in ascending order

>>>list1 = [23,45,11,67,85,56]

>>> list2 = sorted(list1)

>>> list1

[23, 45, 11, 67, 85, 56]

>>> list2

[11, 23, 45, 56, 67, 85]

min()

Returns minimum or smallest element of the list.

max ()

Returns maximum or largest element of the list

sum ()

Returns sum of the elements of the list

>>> list1 = [34,12,63,39,92,44]

>>> **min**(list1)

12

>>> **max**(list1)

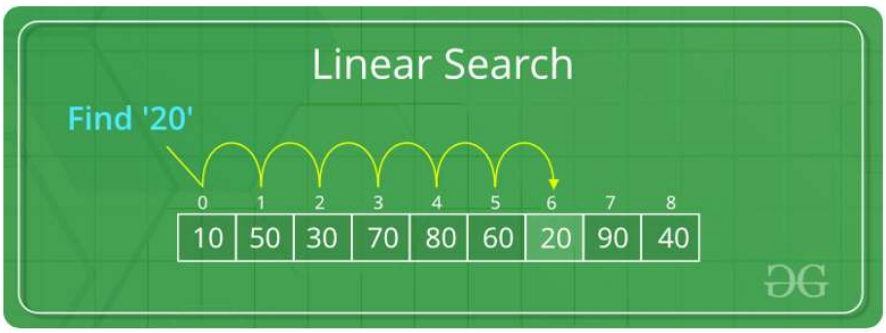
92

>>> **sum**(list1)

28

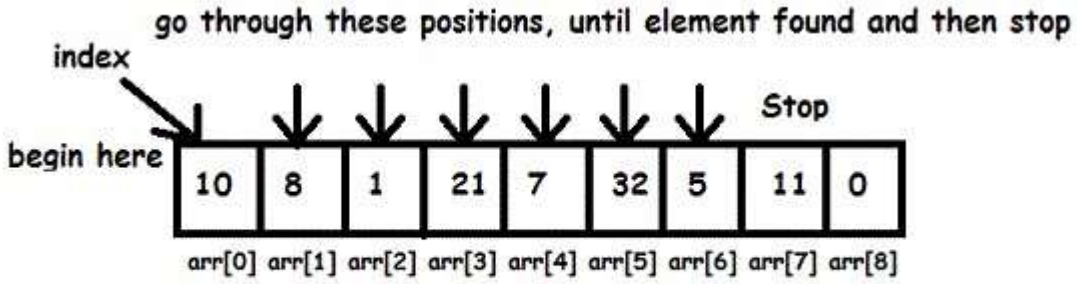
LINEAR SEARCH

A simple approach is to do a **linear search**, i.e
•Start from the leftmost element of arr[] and one by one compare x with each element of arr[]
•If x matches with an element, return the index.



list1= [10,20,30,40,10,50,10]

```
s=int(input("enter the number to be searched:"))
a=len(list1)
for i in range(a):
    if s==list1[i]:
        print("element found at index %d"%i)
        break
else:
    print("element not found")
```



Element to search : 5

OR

15. Write a program to perform linear search on the given list:
[10, 51, 2, 18, 4, 31, 13, 5, 23, 64, 29]

Ans.

```
#Linear Search Implementation
num = [10,51,2,18,4,31,13,5,23,64,29]
pos = 0
print("List elements are: ",end=' ')
for i in num:
    print(i,end=' ')
print()
find = int(input("Enter the element to search: "))
flag = 0
for i in num:
    if(i == find):
        flag = 1
        pos = num.index(i)
        break
if flag == 1:
    print("Element found at index:",pos)
else:
    print("Element not found")
```

DIFFERENCE BETWEEN APPEND AND EXTEND

append () allows to add only 1 items to a list, extend () can add multiple items to a list.

```
>>> m1= [1,2,3,4]
>>> m2= [100,200]
>>> m1. append (5)
>>> m1
[1, 2, 3, 4, 5]
>>> m1. append (6,7)
Traceback (most recent call last):
File "<pyshell#16>", line 1, in <module> m1. append (6,7)
Type Error: append () takes exactly one argument (2 given)
>>> m1.append([6,7])
>>> m1
[1, 2, 3, 4, 5, [6, 7]]
>>> len(m1)
 6
```

Now let us see extend() function

```
>>> m2
 [100, 200]
>>> m2. extend(300)
Traceback (most recent call last):
File "<pyshell#21>", line 1, in <module>
m2. extend (300)
Type Error: 'int' object is not iterable
>>> m2.extend([300,400])
>>> m2
[100, 200, 300, 400]
>>> len(m2)  4
```

PROGRAMS

Program1 : A program to calculate average marks of n students where n is entered by the user.

```
#create an empty list
list1 = []
print("How many students marks you want to enter: ")
n = int(input ())
for i in range(0,n):
print("Enter marks of student",(i+1),":")
marks = int (input ())
#append marks in the list
list1.append(marks)
#initialize total
total = 0
for marks in list1:
        total = total + marks
average = total / n
print ("Average marks of", n,"students is:", average)
```

Program 2: Write a program to check if a number is present in the list or not. If the number is present, print the position of the number. Print an appropriate message if the number is not present in the list.

```
list1 = [] #Create an empty list
print("How many numbers do you want to enter in the list: ")
maximum = int(input())
print("Enter a list of numbers: ")
for i in range (0, maximum):
        n = int(input ())
        list1.append(n) #append numbers to the list
num = int(input("Enter the number to be searched: "))
position = -1
for i in range (0, len(list1)
```

```
    if list1[i] == num: #number is present
        position = i+1 #save the position of number
    if position == -1 :
        print ("Number", num,"is not present in the list")
    else:
        print("Number",num,"is present at", position + 1, "position")
```

Program 3: Write a program to allow user to perform any of the list operations given in a menu. The menu is:

1. Append an element
2. Insert an element
3. Append a list to the given list
4. Modify an existing element
5. Delete an existing element from its position
6. Delete an existing element with a given value
7. Sort the list in the ascending order
8. Sort the list in descending order
9. Display the list.

```
#Program 4-1
#Menu driven program to do various list operations
myList = [22,4,16,38,13] #myList having 5 elements
choice = 0
for attempt in range (3):
    print ("Attempt number:", attempt)
    print("The list 'myList' has the following elements", myList)
    print("\n L I S T O P E R A T I O N S")
    print(" 1. Append an element")
```

```
print(" 2. Insert an element at the desired position")
print(" 3. Append a list to the given list")
print(" 4. Modify an existing element")
print(" 5. Delete an existing element by its position")
print(" 6. Delete an existing element by its value")
print(" 7. Sort the list in ascending order")

print (" 8. Sort the list in descending order")
print (" 9. Display the list")
choice = int (input ("ENTER YOUR CHOICE (1-9): "))
#append element
if choice == 1:
    element = eval (input ("Enter the element to be appended: "))
    myList.append(element)
    print ("The element has been appended\n")
#insert an element at desired position
elif choice == 2:
    element = eval (input ("Enter the element to be inserted: "))
    pos = int(input("Enter the position:"))
    myList.insert(pos,element)
    print("The element has been inserted\n")
#append a list to the given list
elif choice == 3:
    newList = eval(input("Enter the list to be appended: "))
    myList.extend(newList)
    print ("The list has been appended\n")
#modify an existing element
elif choice == 4:
    i = int (input ("Enter the position of the element to be modified: "))
    if i < len(myList):
        newElement = eval (input ("Enter the new element: "))
        oldElement = myList[i]
```

```
            myList[i] = newElement
            print("The element",oldElement,"has been modified\n")
        else:
            print ("Position of the element is more than the length of list")
    #delete an existing element by value
    elif choice == 6:
            element = int(input("\n Enter the element to be deleted: "))
            if element in myList:
                myList.remove(element)
                print ("\n The element", element,"has been deleted\n")
            else:
                print ("\n Element", element, "is not present in the list")
    #list in sorted order
    elif choice == 7:
            myList.sort()
            print("\n The list has been sorted")
    #list in reverse sorted order
    elif choice == 8:
            myList.sort(reverse = True)
            print("\n The list has been sorted in reverse order")
    #display the list
    #display the list
    elif choice == 9:
            print("\n The list is:", myList)
    else:
            print("Choice is not valid")
```

Other programs

- finding the maximum, minimum, mean of numeric values stored in a list
- linear search on list of numbers and counting the frequency of elements in a list