Day Objectives

- Python Data Structure
- Lists
- Tuples
- Dictionaries
- Basic Problem set on Data Structures
- Advanced Problem set

Lists:

Stores any kind of data irrespective of type,it allows to manipulates the data (CRUD operations on database like)

Creating a Lists

In [61]:

```
li = [123,234,456]

li # Access the entire list

li[1] # Access the specific element ,note: index is starts with 0

li[1:]# Access the elements from second position

li[::-1]#not reorder the original list

li = li[::-1] # Original list gets updated

li+li

li = li * 10

li[::2] # Accessing even index elements

li = [123,234,456]

li[1::2] # Accedding odd index elements

li.append(678)

li
```

Out[61]:

```
[123, 234, 456, 678]
```

Lists accessing

```
# Direct referencing
# In direct referencing
```

Direct referencing:

- chance of damage the data
- using [index]

Indirect referencing:

- done through functions
- there are some predefined functions to do

In [62]:

```
li.append(890) # appending the element in a last position
li
```

Out[62]:

```
[123, 234, 456, 678, 890]
```

In [63]:

```
# Adding the element at a specific position
li.insert(1,654)

li
```

Out[63]:

```
[123, 654, 234, 456, 678, 890]
```

In [64]:

```
# Sorting of a elements in a list

li.sort() # sorts acending order
li
```

Out[64]:

```
[123, 234, 456, 654, 678, 890]
```

In [65]:

```
li.insert(1,200)
li
```

Out[65]:

```
[123, 200, 234, 456, 654, 678, 890]
```

In [67]:

```
# remove the last element in a last index

li.pop()
li
```

Out[67]:

```
[123, 200, 234, 456, 654]
```

In [68]:

```
# Remove the at particulat element in a list

li.pop(1)

li
```

Out[68]:

```
[123, 234, 456, 654]
```

In [71]:

```
li2 = [345,456,567]
li2
```

Out[71]:

```
[345, 456, 567]
```

In [73]:

```
# Merge list 2 into list 1

li.extend(li2)
li
```

Out[73]:

```
[123, 234, 456, 654, 345, 456, 567, 345, 456, 567]
```

In [77]:

```
l=[1,2,3,4]
l2=[9,7,6,5]
l.extend(li2)
li
```

Out[77]:

```
[123, 234, 456, 654, 345, 456, 567, 345, 456, 567]
```

In [84]:

```
l
```

Out[84]:

```
[1, 2, 3, 4]
```

In [ ]:

```
# Sum of all elements in a list
sum(l)
```

In [85]:

```
# Max element in a list
max(l)
```

Out[85]:

```
4
```

In [86]:

```
# Len of an element
len(l)
```

Out[86]:

```
4
```

In [88]:

```
# average of elements in a list
sum(l)/len(l)
```

Out[88]:

```
2
```

In [96]:

```
# Average of all alternate elements at even positions

sum(l[::2])/len(l[::2])
```

Out[96]:

```
2.0
```

In [127]:

```
# Average of all alternate elements at odd positions

sum(l[1::2])//len(l[1::2])

l.insert(3,4)
l = [1,3,2,4]
l
```

Out[127]:

```
[1, 3, 2, 4]
```

In [162]:

```
# Identify the second largest element in a list

def findLarge(lis,n):
    temp = lis
    res = 0
    for i in range(1,len(temp)+1):
        if(i <= n):
            print(i)
            lis.pop()
            res = max(temp)

    return res

lis = [3,4,2,6,7]
findLarge(lis,2)
```

```
1
2
```

Out[162]:

```
4
```

In [121]:

```
1
```

Out[121]:

```
[1, 2, 3]
```

In [119]:

```
min(l)
```

Out[119]:

```
1
```

In [169]:

```
# Chaitanya logic

def logic2(listt):
    length = len(listt)
    listt.sort()
    print(listt[length-2])
```

In [170]:

```
lis
```

Out[170]:

```
[3, 4, 2]
```

In [171]:

```
logic2([2, 3, 4, 6])
```

```
4
```

In [177]:

```
# Second Problem:
# Function that returns the n th largest number:
def logicforN(listt,n):
    length = len(listt)
    if(n<=length):
        listt.sort()
        print(listt[length-n]) # listt[-n] also works for this problem
    else:
        print("Given number is exceeded the len")
```

In [183]:

```
logicforN([3,4,5,6,3,6,7,8],7)
```

```
3
```

Function to search for a values

In [6]:

```
# Function to search for data in a list

def checkValue(list,key):
    return True if key in list else False
checkValue(lis,44)
```

Out[6]:

```
False
```

In [206]:

```
# Function to search for

def checkValue2(list,key):
    for i in range(0,len(list)):
        if(key == list[i]):
            print("value is available at",i+1,"position",list[i])

checkValue2(lis,6)
```

```
value is available at 8 position 6
```

In [22]:

```
# Single line solution for this
lis = [1,2,3,4,5,8,7,6,6]
def sl2(li,k):
    return li.index(k)
sl2(lis,2)
```

Out[22]:

```
1
```

In [16]:

```
# MY Solution
def sl3(li,key):
    return li.index(key) if key in li else -1
```

In [19]:

```
sl3(lis,3)
```

Out[19]:

```
2
```

In [23]:

```
lis.count(6)
```

Out[23]:

```
2
```

In [24]:

```
# Function to count the occurences of a

value = "Python programming"
value.count("m")
```

Out[24]:

```
2
```

In [40]:

```
# Function to find the no of occurances of a substring

def countSubString(str,sub_str):
    for i in range(0,len(sub_str)):
        if
```

In [43]:

```
n = int(input("Enter the size of the array:"))
s = input("Enter elements : ")
li = s.split()
con_list = []
for i in li:
    con_list.append(int(i))
con_list
```

```
Enter the size of the array:5
Enter elements : 2 3 4 5 6 7
```

Out[43]:

```
[2, 3, 4, 5, 6, 7]
```