Problem Solving and Programming in Python Day-4

Date - 14 June 2019

Day Objectives

- Python Data Structures
  - Lists
  - Tuples
  - Dictionaries
- Dictionaries
- Advanced Problem Set
- Packages and Modules in Python

Python Data Structures

Lists

In [49]:

```
li = [123, 978, 654]

li # Access the entire list

li[1] # Access an element with index in a list

li[1:] # Access all elements second element to last element

li[::-1] # copying the elements and give us in reverse order but still we have t

li # same list is there by above reverse order

li = li[::-1]  # donot copying putting entire list as in reverse order

li # list has changed as reverse order

li = li[::-1]

li[::2] # Accessing even index elements as a list

li[1::2] # Accessing odd index elements as a list

li[1] # Accessing the index value not as list but as value  i.e we donot get ""[

# Lists can be accessed , manipulated in two Different ways
    # Direct referencing - [index] # Accessing using  ""[]"" square brackets
    # Indirect Referencing   - through functions

li.append(345)  # Adding an element to end of the list

li

li.insert(1,234)  # Adding an element at a particular position(index)

li

li.sort()  # Sort elements in ascending order

li

li.pop()   # Remove the last element in a list  and return it

li  # Showing the list after removing the last elements i.e [123, 234, 345, 654]

li.pop(1) # Remove an element at a particular index

li   # Showing the list after removing 1 index element i.e 234 op = [123,345,654

li2 = [234,456,789]

li.extend(li2)# Merge list 2 into list 1 i.e li2 elements added to li

li   # o/p --> [123, 345, 654, 234, 456, 789]

sum(li)    #  SUM of all elements in a list (if elements are all numbers)
```

```
max(li)  # Maximum element in a list

len(li)  # Number of elements in a list

sum(li)/len(li)  # Average of list elements

sum(li[::2])/len(li[::2])   # Average of all alternate elements at even positions


sum(li[1::2])/len(li[1::2])  # Average of all alternate elements at odd positions

li

min(li)   # Minimum element in a list
li

min(li)-1   #[123, 345, 654, 234, 456, 789]    --> op-->122

try:
    li.index(1000)
except:
    print(-1)
```

```
-1
```

In [3]:

```
n = int(input())
s = [input()]
li = s
li
```

```
3
 1 2 3
```

Out[3]:

```
[' 1 2 3 ']
```

In [4]:

```
s = " 1 2 3 4 5 6"
li = s.split()
numberlist = []
for i in li:
    numberlist.append(int(i))
numberlist
```

Out[4]:

```
[1, 2, 3, 4, 5, 6]
```

In [43]:

```
# Function to identify the second largest elements in a unique list
    # Sort the data and select the second last elements
    # Sort the data in reverse order, and select the max
    # Remove the max element and then get the max of the new list
    #

def secondLargest(li):
    li.sort()
    return li[-2]


# Function that returns the nth largest

def genericLargest(li, n):
    li.sort()
    return li[-n]
secondLargest(li)
genericLargest(li, 5)
```

Out[43]:

```
234
```

In [51]:

```
# Function to search for data in a list
# Search for the key in the list and return the index of the key.return -1 if ke

def linearSearch(li, key):
    for index in range(0,len(li)):    # for value in li:
        if li[index] == key:
            return index    # return index+1
    return -1

def linearSearch2(li, key):
    for element in li:
        if element == key:
            return li.index(element)
    return -1

def linearSearch3(li, key):
    if key in li:
        return li.index(key)
    return -1
                                            #return index(key) --> index of that
linearSearch1(li, 234)
linearSearch2(li, 234)
linearSearch3(li, 234)
```

Out[51]:

```
3
```

In [57]:

```
# Function to count the occurances of a character in a string
# "Python Programming", m -> 2
def countCharOccurances1(string,character):
    count = 0
    for ch in string:
        if ch == character:
            count += 1
    return count

def countCharOccurances2(s,c):
    return s.count(c)

countCharOccurances1("Python Programming", "m")

countCharOccurances2("Python Programming Py", "Py")

# Function to find the number of occurances of a substring
# "abcabcddcba", "ab"  --> 2

def countSubString(s,c):
    l=len(c)
    count=0
    for  hd in s:
```

Out[57]:

```
2
```

In [2]:

```
n = int(input())

def sumofSqauresNaturalNumbers(n):
    sum = 0
    for i in range(1, n+1):
        sum = sum + i ** 2
    return sum
sumofSqauresNaturalNumbers(n)
```

```
3
```

Out[2]:

```
14
```

In [ ]: