Problem Solving and Programming in Python

Date -14 June 2019

Day Objectives

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

Python Data Structure

Lists

In [2]:
```
li=[123,978,654]

li # Access the entire list

li[1] # Accessing a particular element with index in list

li[1:] # Access all element from second to last(slicing)

li[-1::-1] # Accessing all the elements of the list in reverse order.

li= li[-1::-1]

li

li= li[-1::-1] # reversing the list and reassining to original list

li

li[::2] # Accessing even index elements

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

# List can be accessed and manipulated in two different ways
        # Direct Referencing --> [index]
        # Indirect Referencing --> functions

# Adding an element to the end of list

# Indirect Referencing
li.append(6)

li

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

li.sort() # Sorts all the elements of the list in the ascending order

li

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

li

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

li

li2=[3,4,5]

li.extend(li2) # Merge list2 into list1

li

sum(li) # Caculates the sum of all the elements in the list

max(li) # Return the maximum element in the list
```

```
len(li) # Return the length of the list

sum(li[0::2])/len(li[0::2]) # Average of elements at even position

sum(li[1::2])/len(li[1::2]) # Average of elements at odd position
```

Out[2]: 80.66666666666667

Average of elements in a list

In [3]:
```
l=[1,2,3,4,5,6]
s=sum(l)
c=len(l)
avg=s/c
print(avg)
```

```
3.5
```

Find the second largest element in the list

In [4]:
```
# Sort the data and select the second last element
# Sort the data in reverse order and select the second element
# Remove the max element and then get the max

def secondLargest(li):
    li.sort()
    li.reverse()
    return li[1]

secondLargest([1,2,3,5,4])

# Alternate solution--> 1
def seLar(li):
    li.sort()
    return li[-2]

seLar([12,4,5])
```

Out[4]: 5

Function that returns the nth largest

In [5]:
```
def genericLargest(l,n):
    l.sort()
    return l[-n]

genericLargest(l,3)
```

Out[5]: 4

Function to search for an element in a list

Search for the key in the list and return the index of the key. Return -1 if the key is not found

In [6]:
```
li=[3,4,5,6,234,654]
def LinearSearch(li,key):
    for i in li:
        if li[i] == key:
            return i
    return -1
#LinearSearch(li,3)

def LinearSearch2(li,key):
    for element in li:
        if element == key:
            return li.index(element)
    return -1
#LinearSearch2(li,654)
#li
def LinearSearch3(li,key):
    if key in li:
        return li.index(key)
    return -1
LinearSearch3(li,654)
```

Out[6]: 5

Function to count the occurences of a character in a string.

In [7]:
```
def count(s,k):
    return s.count(k)
count('Python Programming','m')

#Alternate solution:
def count2(s,k):
    count = 0
    for ch in s:
        if ch == k:
            count += 1
    return count
count2('Python Programming','m')
```

Out[7]: 2

Function to count the occurences of a given substring in a string

In [8]:
```
def count3(s,k):
    return s.count(k)
count3("Python Python","Pyth")

# Alternate solution:
def count4(s,k):
    d=0
    i=0
    c1=len(k)
    c2=len(s)
    while(i<c2):
        if s[i:i+c1]==k[0:c1]:
            d=d+1
        i+=1
    return d
count4("abaabdaaannaa","aa")
```

Out[8]: 4

Write a program to find sum of squares of a given n numbers

In [9]:
```
N=int(input())
sum=0
for i in range(1,N+1):
    if i<=N:
        i = i ** 2
        sum+=i
print(sum)
```

```
3
14
```

Closest to zero

Explaination

- list of numbers li=[3,2,-1,-2,-3] (Original List)
- Sort the data li.sort()
- li = [-3,-2,-1,2,3] (Sorted List)
- pl = [1,2,2,3,3] (Positive Sorted List)
- pl[0] -> Check if this number is negative or positive in original list
- if pl[0] in li:
  - return pl[0]
- else
  - return -pl[0]

In [10]:
```
li=[-1,1,-2,2,3]

li.sort()

pl = []

for i in li:
    pl.append(abs(i))

pl.sort()

if pl[0] in li:
    print (pl[0])

else:
    print (-pl[0])
```

```
1
```

In [11]:
```
li=[-1,1-2,2,3]

li.sort()

pl = []

for i in li:
    pl.append(abs(i))

pl.sort()

if -pl[0] in li:
    print (-pl[0])

else:
    print (pl[0])
```

```
-1
```

Farthest from zero

In [12]:
```
li =[-1,-2,1,-10,-9]

li.sort()

pl=[]
for i in li:
    pl.append(abs(i))

pl.sort()

if pl[-1] in li:
    print(pl[-1])

else:
    print(-pl[-1])
```

```
-10
```

Problem - 3 (HackerEarth)

You are given three numbers a,b and c. Write a program to find the largest number which is less than or equal to c and leaves remainder b when divided by a.

- 3 2 9 --> 8
- 9 % 3 == 0
- 8 % 3 == 2
- Output : 8
- 4 % 1 == 0
- 3 % 1 == 0
- 2 % 1 == 0
- 1 % 1 == 0
- 0 % 1 == 0
- -1 % 1 == 0

In [16]:
```
def cal(a,b,c):
    for i in range(c,a-1,-1):
        if i % a == b:
            return i
    return -1
cal(3,2,100)
```

Out[16]:
```
98
```

In [ ]:
```

```