**16/2/2019**

**10/02 Assignments Revision**

**String Length**

**Simple**

count=0

while inputString[count]!=0

count+=1

**Recursion**

def LengthString(inputString)

if inputString==’’

return 0

return 1+LenthString(inputString[1:])

Number of 1’s

**Simple**

def countOne(no)

count=0

while(no!=0)

count+=1

no=no&(no-1)

**Recursion**

def countOne(inpuNo)

if(inputNo==0)

return 0

return 1+countOne(inputNo&(inputNo-1))

**Containers in PYTHON**

**List : - Collection of heterogenius elements,**

**- mutable containers**

**- can be changed.**

**>>> x=[1,2,'x',4]**

**>>> x**

**[1, 2, 'x', 4]**

**>>> type(x)**

**<type 'list'>**

**>>> x.append('j')**

**>>> x**

**[1, 2, 'x', 4, 'j']**

**>>> x.append([4,5,6])**

**>>> x**

**[1, 2, 'x', 4, 'j', [4, 5, 6]]**

**>>> x.extend([4,5,6])**

**>>> x**

**[1, 2, 'x', 4, 'j', [4, 5, 6], 4, 5, 6]**

**>>> x.insert(1,9)**

**>>> x**

**[1, 9, 2, 'x', 4, 'j', [4, 5, 6], 4, 5, 6]**

**>>> x[::-1]**

**[6, 5, 4, [4, 5, 6], 'j', 4, 'x', 2, 9, 1]**

**>>> x.pop()**

**6**

**>>> x**

**[1, 9, 2, 'x', 4, 'j', [4, 5, 6], 4, 5]**

**>>> x.pop(5)**

**'j'**

**>>> x**

**[1, 9, 2, 'x', 4, [4, 5, 6], 4, 5]**

**>>>>>> x.remove(4)**

**>>> x**

**[1, 9, 2, 'x', [4, 5, 6], 4, 5]**

**>>>**

**#WAP to remove all occuerances of given element from list**

#wap to remove all occurances of character from String

def RemoveOccurance(inputList,characterInput):

while characterInput in inputList:

inputList.remove(characterInput)

def main():

inputList=input("Enter List : ")

characterInput=input("Enter character to remove: ")

RemoveOccurance(inputList,characterInput)

print(inputList)

if \_\_name\_\_=='\_\_main\_\_':

main()

**#WAP to implement Stack operations on list**

**#Stack implementation**

def StackOperation(inputList,operation):

if operation=='pop':

inputList.remove(inputList[-1])

elif operation=='push':

element=input("Enter element to push")

inputList.append(element)

elif operation=='peep':

inputList[-1]

elif operation=='isEmpty':

if len(inputList)==0:

print("Stack is empty")

else:

print("Stack is not empty")

elif operation=='isFull':

if len(inputList)==10:

print("Stack is Full")

else:

print("Stack is not full yet")

def main():

inputList=input("Enter List")

operation=input("Enter operation")

StackOperation(inputList,operation)

print(inputList)

if \_\_name\_\_=='\_\_main\_\_':

main()

**#write menu driven for Stack operations on list.**

**#WAP to demonstrate Queue data structure with following :**

**enqueue,dequeue,isEmpty,isFull**

**#WAP to accept list of integers from user and check if it is sorted or not**

def CheckIsSorted(inputList):

index=0

while(index!=len(inputList)-1):

if(inputList[index]<inputList[index+1]):

index+=1

else:

index+=1

print("List is not sorted")

return False

print("List is sorted")

def main():

inputList=input("Enter List")

CheckIsSorted(inputList)

if \_\_name\_\_=='\_\_main\_\_':

main()

**#WAP to accept unsorted list from user and sort it**

#WAP to sort given list

def SelectionSort(inputList):

for i in range(len(inputList)-1):

for j in range(i+1,len(inputList)):

if(inputList[i]>inputList[j]):

temp=inputList[i]

inputList[i]=inputList[j]

inputList[j]=temp

return inputList

def main():

inputList=input("Enter List")

print(SelectionSort(inputList))

if \_\_name\_\_=='\_\_main\_\_':

main()

**#HW implement Bubble sort**

hint: compare 1->2, 2->3,3->4 and swap

**#HW implement Insertion sort**

hint : take element and compare with whole stack and then place it