University Of Baltistan

Submitted By: **Basit Ali**

Submitted to **Maam Noreen**

RegNo: **S23BSCS012**

Assigment: **02 to 08**

Date: **04/04/2024**

ASSIGMENT 02:

Q1: Modify the example #6 lab#2, insert at least three new elements e1,e2,e3 at index 0,2,4 respectively?

```
from array import *
a1=array('i',[23,56,12,14,5])
index0=0
index1=2
index2=4
e1=55
e2=65
e3=75
a1.insert(index0,e1)
a1.insert(index1,e2)
a1.insert(index2,e3)
for i in a1:
    print(i,end=" ")
```

Output Of The above program is:

```
55 23 65 56 75 12 14 5
PS C:\Users\hp\Saved Games\OneDrive\Desktop\DSAUNIS>
```

Q2: consider an array[33,44,55,33,80] the remove the element 33 from the list. Your resultant array should be [44,55,80]?

```
from array import *
a2=array('i',[33,44,55,33,80])
a2.pop(0)
a2.pop(2)
for i in a2:
    print(i,end=" ")
```

Output of the above Program:

```
44 55 80
PS C:\Users\hp\Saved Games\OneDrive\Desktop\DSAUNIS>
```

ASSIGMENT 03:

Q1: Write a program to sort a list of elements in reverse order (descending order)? And

Q2: Concatenate at least three lists by using extends function?

```
# ------------------>ASSIGMENT 02<---------------------
# 1 Write a program to sort a list of elements in reverse order(descending
order)
# Ans1:
myList=[23,16,31,20,56,7,99,11,9]
myList.sort(reverse=True)
print("#1: Sorted output of a list of elements in reverse order ", myList)

# 2 Concatenate at least three lists by using extends funcion
myList1=[1,2,3]
myList2=['a','b','c']
myList3=[4,5,6,7]
myList2.extend(myList3)
myList1.extend(myList2)
print("#2: Generat list by extend function" ,myList1)
```

Output of Above Program:

```
#1:Sorted output of a list of elements in reverse order
 [99, 56, 31, 23, 20, 16, 11, 9, 7]
#2:Generated list by extend function
 [1, 2, 3, 'a', 'b', 'c', 4, 5, 6, 7]
PS C:\Users\hp\Saved Games\OneDrive\Desktop\DSAUNIS>
```

Assigment 04:

Q1: Repeat the example 3 lab4 without using the init() function?

```python
class Teacher:
    def setInformation(self, teachername=None, teachersubject=None,teachercity=None):
        self.teachername = teachername
        self.teachersubject = teachersubject
        self.teachercity = teachercity
    def set_tname(self, teachername):
        self.teachername = teachername
    def set_tsub(self, teachersubject):
        self.teachersubject = teachersubject
    def set_tcity(self, teachercity):
        self.teachercity = teachercity
    def get_tname(self):
        return self.teachername
    def get_tsub(self):
        return self.teachersubject
    def get_tcity(self):
        return self.teachercity
```



```python
# Create instances of teachers
maam_saima = Teacher()
maam_noreen = Teacher()
sir_jawad = Teacher()
# Set teacher information using getInformation
maam_saima.setInformation('Saima', 'Linear Algebra', 'Skardu')
maam_noreen.setInformation('Noreen', 'DSA', 'Skardu')
sir_jawad.setInformation('Jawad Usman', 'OOP', 'Skardu')
# Print information
print(f"{maam_saima.get_tname()} teaches {maam_saima.get_tsub()} in {maam_saima.
get_tcity()}")
print(f"{maam_noreen.get_tname()} teaches {maam_noreen.get_tsub()} in {maam_noreen.
get_tcity()}")
print(f"{sir_jawad.get_tname()} teaches {sir_jawad.get_tsub()} in {sir_jawad.
get_tcity()}")
```

Output of the above Program:

```
Saima teaches Linear Algebra in Skardu
Noreen teaches DSA in Skardu
Jawad Usman teaches OOP in Skardu
PS C:\Users\hp\Saved Games\OneDrive\Desktop\DSAUNIS>
```

ASSIGNMENT 05:

Q1: Modify the example #1 and #2 lab #5 by inserting three items roll_no,name and cgpa in the data part. Compile both of the example 1 and 2 as a one program?

```
class Node:
    def __init__(self,Std_reg=None,Std_name=None,Std_cgpa=None,next=None):
        self.Std_reg=Std_reg
        self.Std_name=Std_name
        self.Std_cgpa=Std_cgpa
        self.next = next
class LinkedList:
    def __init__(self,start=None):
        self.start=start
    def isEmpty(self):
        return self.start == None
    def insertAtBegining(self,Std_reg,Std_name,Std_cgpa):
        newNode = Node(Std_reg,Std_name,Std_cgpa,self.start)
        self.start=newNode
```

```
    def insertAtLast(self,Std_reg,Std_name,Std_cgpa):
        temp=self.start
        newNode=Node(Std_reg,Std_name,Std_cgpa)
        if self.isEmpty():
            self.start=newNode
        else:
            while temp.next!=None:
                temp=temp.next
            temp.next=newNode
    def printList(self):
        temp=self.start
        while temp is not None:
            print(f'{temp.Std_reg} {temp.Std_name} {temp.Std_cgpa}')
            temp=temp.next
```



```
myList=LinkedList()
myList.insertAtBegining("S23BSCS012","Basit Ali",3)
myList.insertAtBegining("S23BSCS018","Zeeshan Haider",6)
myList.insertAtBegining("S23BSCS024","Rafi Khan",8)
myList.printList()
```

Output Of the above Program is:

```
S23BSCS024 Rafi Khan 8
S23BSCS018 Zeeshan Haider 6
S23BSCS012 Basit Ali 3
PS C:\Users\hp\Saved Games\OneDrive\Desktop\DSAUNIS>
```

ASSIGMENT 06:

Q1: Add new functions insert_At_Sec and Insert_At_Sec_Last in the above program lab#6 exp#1 to add the data at second postion and second last postion in linklist?

```
class Node:
    def __init__(self,data=None,next=None):
        self.data = data
        self.next=next
class LinkList:
    def __init__(self,head=None):
        self.head = head
    def isempty(self):
        return self.head==None
    def insert_at_start(self,data):
        temp=self.head
        obj=Node(data,temp)
        self.head=obj
```

```
 def insert_at_last(self,data):
     temp=self.head
     obj=Node(data)
     if(self.isempty()):
         self.head = obj
     else:
          while(temp.next!=None):
                temp=temp.next
          temp.next=obj
 def insertAfter(self,posi,item):
     if(self.head==None):
         newNode=Node(item)
         self.head=newNode
     else:
         temp=self.head
         while temp.data != posi:
             temp=temp.next
             if(temp==None):
                 break
         if(temp==None):
             print(f'Sorry! Element Not Found')
         else:
             newNode=Node(item,temp.next)
             temp.next=newNode
```

```
    def insertAtSec(self,item):
        temp=self.head
        if(self.isempty()):
            newNode=Node(item)
            self.head=newNode
        else:
            newNode=Node(item,temp.next)
            temp.next=newNode
    def insertAtSecLast(self,item):
        temp=self.head
        prev=None
        while temp.next != None:
            prev=temp
            temp=temp.next
        newNode=Node(item,temp)
        prev.next=newNode
```

```
    def show(self):
        temp=self.head
        while(self.head!=None):
            print(self.head.data, " ")
            self.head=self.head.next
        self.head=temp
# Testing the code
llist=LinkList()
# Inserting elements at the start and last of a list
llist.insert_at_start("B")
llist.insert_at_last("D")
llist.insertAfter("D",7)
llist.insertAtSec("G")
llist.insertAtSecLast(5)
# printing the list
llist.show()
```

Output of program is:

```
Output is :
B G D 5 7
PS C:\Users\hp\Saved Games\OneDrive\Desktop\DSAUNIS>
```

ASSIGMENT 07:

Q1: Write a program to delete the specific element from the linklists?

```python
class Node:
    def __init__(self,data=None,next=None):
        self.data = data
        self.next=next
class LinkList:
    def __init__(self,head=None):
        self.head = head
    def isempty(self):
        return self.head==None
    def insert_at_start(self,data):
        temp=self.head
        obj=Node(data,temp)
        self.head=obj
```

```
    def delAtPosi(self,key):
        if(self.isempty()):
            print("Linked List is Empty")
        else:
            temp=self.head
            prev=None
            self.i=1
            while self.i<key:
                prev=temp
                temp=temp.next
                if(temp==None):
                    break
                self.i+=1
            if(temp==None):
                print("Out Of Range")
            else:
                prev.next=temp.next
```

```
    def del_first(self):
        if(self.isempty()):
            print("Linked List is empty")
        else:
            self.head=self.head.next
    def delSpecificVal(self,item):
        if(self.isempty()):
            print("Lists is Empty")
        else:
            temp=self.head
            temp1=None
            while temp.data!=item:
                temp1=temp
                temp=temp.next
                if(temp==None):
                    break
            if(temp==None):
                print('Element Not Found')
            else:
                temp1.next=temp.next
```

```
    def del_last(self):
        if(self.isempty()):
            print("Linked list is empty")
        elif(self.head.next==None):
            self.head=None
        else:
            temp=self.head
            temp1=None
            while(temp.next!=None):
                temp1=temp
                temp=temp.next
            temp1.next=None
    def show(self):
        temp=self.head
        if(temp==None):
            print('Linklists is Empty!!!!!!!!')
        while(self.head!=None):
            print(self.head.data, " ")
            self.head=self.head.next
        self.head=temp
```

```
# Testing the code
llist=LinkList()
# Inserting elements at the start and last of a list
llist.insert_at_start(99)
llist.insert_at_start(101)
llist.insert_at_start(49)
llist.insert_at_start(453)
llist.insert_at_start(43)
print("Before Deleteing")
llist.show()
llist.del_first()
llist.delSpecificVal(49)
llist.del_last()
print("Before Deleteing First Last and Specfic Value")
# printing the list
llist.show()
```

Output of the Program:

```
Before Deleteing
43
453
49
101
99
Before Deleteing First Last and Specfic Value
453
101
PS C:\Users\hp\Saved Games\OneDrive\Desktop\DSAUNIS>
```

ASSIGMENT 08:

Q1: Write a code to Stack using linklist?

```
class Node:
    def __init__(self, data=None,next=None):
        self.data = data
        self.next = next
class Stack:
    def __init__(self,head=None):
        self.head=head
    def push(self,item):
        newNode = Node(item)
        if not self.head:
            self.head = newNode
        else:
            newNode.next = self.head
            self.head = newNode
    def pop(self):
        if self.is_empty():
            return "Underflow"
        else:
            temp = self.head
            self.head = self.head.next
            return temp.data
    def is_empty(self):
        return self.head == None
    def peek(self):
        if self.is_empty():
            return "No element in the stack"
        else:
            return self.head.data
    def show(self):
        current = self.head
        while current is not None:
            print(current.data)
            current=current.next
Obj=Stack()
Obj.push(1)
Obj.push(3)
Obj.push(5)
Obj.push(58)
print(Obj.peek())
Obj.pop()
Obj.show()
```

Output Of the Program is:

```
value at peek is :  58
5
3
1
PS C:\Users\hp\Saved Games\OneDrive\Desktop\DSAUNIS>
```