NAME: Tausif Khan

SUBJECT: OSTL

STD: SE COMPS(B)

ROLL NO: 12

**Experiment No: 8.**

**AIM:** A Python program to:

1. Menu driven program to create a linked list and perform all the operations on the list.
2. Menu driven program to perform various (all) operations on a Stack.

**Tools Used:** Python 3.4.3, Terminal

**Theory:**

**Explain basics data structures**

1. Handling one or two values is very easy in a program.
2. Eg. to store two values like 10 and “Jack”, we can take two variables and store these values into those variables.
3. This task would be difficult when there are several values. eg. a small company having more than 100 employees whose details should be stored in memory and processed.
4. Python provides data structures like arrays, lists, tuples, sets and dictionaries.
5. These data structures store elements in various models.
6. A data structure represents arrangement of elements in memory in a particular model.

**Explain ADT**

The abstract datatype is special kind of datatype, whose behavior is defined by a set of values and set of operations.

The keyword “Abstract” is used as we can use these datatypes, we can perform different operations.

But how those operations are working that is totally hidden from the user.

The ADT is made of with primitive datatypes, but operation logics are hidden.

Some examples of ADT are Stack, Queue, List etc

**Explain what is linked list (How it is represented using diagram)**

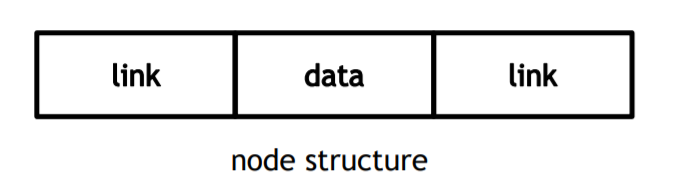
A linked list contains a group of elements in form of nodes.

Each node will have 3 fields:

1. The data field that contains data.
2. A link field that contains reference to the previous node.
3. Another link field that contains reference to the next node.

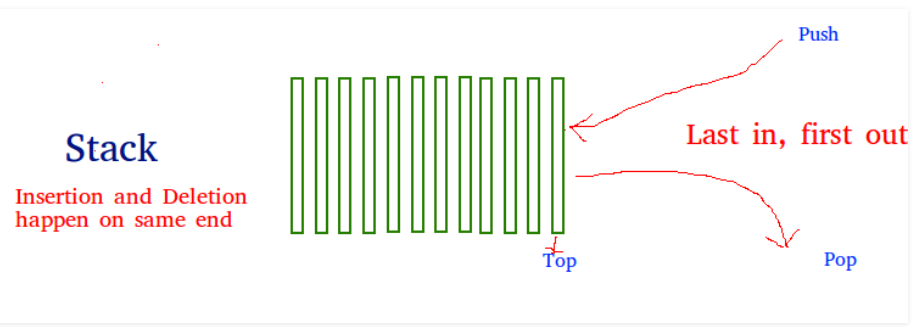
Link fields store references (memory locations).

These link fields are useful to move from one node to another node in the linked list so that any operation can be done in a minimum amount of time.



**Explain what is stack (How it is represented using diagram)**

1. A stack represents a group of elements stores in LIFO (Last In First Out) order.
2. This means that the element which is stored as a last element into the stack will be the first element to be removed from the stack.
3. Inserting elements into stack is called push operation.
4. Removing elements from stack is called pop operation.
5. Searching for an element and returning it without removing it from the stack is called peep operation.
6. Insertion and deletion of elements take place only from one side of the stack, called “top” of the stack



**1. Menu driven program to create a linked list and perform all the operations on the list.**

**CODE:**

l=[]

l.append('Alec')

l.append('Indiana')

l.append('Polis')

op=0

while op<7:

print("1. To add element")

print("2. To remove element")

print("3. To replace element")

print("4. To search an element")

print("5. To insert element at entered position")

print("6 To Display Linked List")

print("7 to exit")

op=int(input("Enter choice "))

if op==1:

x=input("Enter element ")

l.append(x)

elif op==2:

try:

x=input("Enter element to remove ")

l.remove(x)

except ValueError:

print("Element not found")

elif op==3:

newe=input("Enter new element ")

pos=int(input("Enter position "))

l.pop(pos)

l.insert(pos,newe)

elif op==4:

try:

ser=input("Enter element to search ")

pos=l.index(ser)

print("Element found at position ",pos)

except ValueError:

print("Element Not found")

elif op==5:

x=input("Enter element to insert ")

pos=int(input("Enter position to insert "))

l.insert(pos,x)

elif op==6:

print(l)

else:

break

print(l)

**OUTPUT**

1. To add element

2. To remove element

3. To replace element

4. To search an element

5. To insert element at entered position

6 To Display Linked List

7 to exit

Enter choice 6

['Alec', 'Indiana', 'Polis']

1. To add element

2. To remove element

3. To replace element

4. To search an element

5. To insert element at entered position

6 To Display Linked List

7 to exit

Enter choice 1

Enter element Rebound

1. To add element

2. To remove element

3. To replace element

4. To search an element

5. To insert element at entered position

6 To Display Linked List

7 to exit

Enter choice 6

['Alec', 'Indiana', 'Polis', 'Rebound']

1. To add element

2. To remove element

3. To replace element

4. To search an element

5. To insert element at entered position

6 To Display Linked List

7 to exit

Enter choice 2

Enter element to remove Indiana

1. To add element

2. To remove element

3. To replace element

4. To search an element

5. To insert element at entered position

6 To Display Linked List

7 to exit

Enter choice 6

['Alec', 'Polis', 'Rebound']

1. To add element

2. To remove element

3. To replace element

4. To search an element

5. To insert element at entered position

6 To Display Linked List

7 to exit

Enter choice 7

['Alec', 'Polis', 'Rebound']

**2. Menu driven program to perform various (all) operations on a Stack.**

**CODE:**

**#STACK MODULE**

class Stack:  
 def \_\_init\_\_(self):  
 self.stack=[]  
 def emptychk(self):  
 return self.stack==[]  
 def push(self,element):  
 self.stack.append(element)  
 def pop(self):  
 if self.emptychk():  
 return -1  
 else:  
 return self.stack.pop()  
 def peep(self):  
 n=len(self.stack)  
 return self.stack[n-1]  
 def search(self,element):  
 if self.emptychk():  
 return -1  
 else:  
 try:  
 n=self.stack.index(element)  
 return len(self.stack)-n  
 except ValueError:  
 return -2  
 def display(self):  
 return self.stack

**#STACK MAIN FILE**

from Stackmod import Stack

obj=Stack()

ch=0

while ch<6:

print("1. Push")

print("2. Pop")

print("3. Peep")

print("4. Search For Element")

print("5. Display")

print("6. Exit")

ch=int(input("Enter choice "))

if ch==1:

x=int(input("Enter element "))

obj.push(x)

elif ch==2:

element=obj.pop()

if element==-1:

print("Stack is empty ")

else:

print("Element popped is ",element)

elif ch==3:

x=obj.peep()

print("Element at stack top is ",x)

elif ch==4:

x=int(input("Enter element "))

pos=obj.search(x)

if pos==-1:

print("Stack is empty")

elif pos==-2:

print("Element not found in the stack")

else:

print("Element found at position ",pos)

elif ch==5:

print('Stack : ',obj.display())

else:

break

print("Stack ",obj.display())

**OUTPUT**

1. Push

2. Pop

3. Peep

4. Search For Element

5. Display

6. Exit

Enter choice 1

Enter element 85

1. Push

2. Pop

3. Peep

4. Search For Element

5. Display

6. Exit

Enter choice 1

Enter element 35

1. Push

2. Pop

3. Peep

4. Search For Element

5. Display

6. Exit

Enter choice 1

Enter element 59

1. Push

2. Pop

3. Peep

4. Search For Element

5. Display

6. Exit

Enter choice 1

Enter element 55

1. Push

2. Pop

3. Peep

4. Search For Element

5. Display

6. Exit

Enter choice 1

Enter element 88

1. Push

2. Pop

3. Peep

4. Search For Element

5. Display

6. Exit

Enter choice 2

Element popped is 88

1. Push

2. Pop

3. Peep

4. Search For Element

5. Display

6. Exit

Enter choice 3

Element at stack top is 55

1. Push

2. Pop

3. Peep

4. Search For Element

5. Display

6. Exit

Enter choice 4

Enter element 35

Element found at position 3

1. Push

2. Pop

3. Peep

4. Search For Element

5. Display

6. Exit

Enter choice 5

Stack : [ 85, 35, 59, 55]

1. Push

2. Pop

3. Peep

4. Search For Element

5. Display

6. Exit

Enter choice 6

Stack [ 85, 35, 59, 55]

**CONCLUSION:**

Thus we have successfully studied and implemented Linked List and Stack Data Structure in Python.