**Python**

Om Ghanshyam Bhamare

Div: A Roll No: 3

SE

**Experiment 5**

**Aim:** Menu driven program for data structure using built in function for queue, dequeue, stack and linked list

**Code:**

*#Om Ghanshyam Bhamare*

*#Div A Roll No: 3*

*#AIM: Menu driven program for data structure using built in function for*

*#queue, dequeue, stack and linked list.*

*import* collections

*def* *repetStack*(lst):

    print("\n\nSelect Stack Operations: ")

    print("\n1]Add Element\n2]Remove Element\n3]Search\n4]Peep\n5]Show")

    choiceStack*=*int(input("Enter Choice: "))

*if* choiceStack*==*1:

        addStack(lst)

*elif* choiceStack*==*2:

        removeStack(lst)

*elif* choiceStack*==*3:

        serachStack(lst)

*elif* choiceStack*==*4:

        peepStack(lst)

*elif* choiceStack*==*5:

        showStack(lst)

*else*:

        print("Invalid choice")

        repetStack(lst)

*def* *addStack*(lst):

    num *=* int(input("\nEnter Element added to stack: "))***;***

    lst.append(num)

    repetStack(lst)

*def* *removeStack*(lst):

    lst.pop()

    repetStack(lst)

*def* *serachStack*(lst):

    num*=*int(input("\nEnter Number to search: "))

*if* num *in* range(len(lst)):

        print("Number is present at index: {0}".format(lst.index(num)))

*else*:

        print("Number is not present")

    repetStack(lst)

*def* *peepStack*(lst):

    print("Element at top : ",lst[*-*1])

    repetStack(lst)

*def* *showStack*(lst):

    print("Elements in Stack : ",lst)

    repetStack(lst)

*def* *stackOp*():

    print("\nStack Operations")

    lst*=*[]

    size*=*int(input("Enter Size of Stack: "))

    print("Enter Element in Stack")

*for* i *in* range(0,size):

        a*=*int(input())

        lst.append(a)

    print("Element in stack: ",lst)

    repetStack(lst)

*#Queue Operations....*

*def* *repetQueue*(lst):

    print("\n\nSelect Queue Operations: ")

    print("\n1]Add Element\n2]Remove Element\n3]Search\n4]Show")

    choiceQueue*=*int(input("Enter Choice: "))

*if* choiceQueue*==*1:

        addEndQueue(lst)

*elif* choiceQueue*==*2:

        removeQueue(lst)

*elif* choiceQueue*==*3:

        serachQueue(lst)

*elif* choiceQueue*==*4:

        showQueue(lst)

*else*:

        print("Invalid choice")

        repetQueue(lst)

*def* *addEndQueue*(lst):

    num*=*int(input("\nEnter element to enter to Queue: "))

    lst.append(num)

    repetQueue(lst)

*def* *removeQueue*(lst):

*if* lst*==*[]:

        print("\nQueue is empty you can not remove elemnt....")

*else*:

        lst.pop(0)

    repetQueue(lst)

*def* *serachQueue*(lst):

    num*=*int(input("\nEnter Number to search: "))

*if* num *in* range(len(lst)):

        print("\nNumber is present at index: {0}".format(lst.index(num)))

*else*:

        print("\nNumber is not present")

    repetQueue(lst)

*pass*

*def* *showQueue*(lst):

    print("\nElements in Queue are: ",lst)

    repetQueue(lst)

*pass*

*def* *queueOp*():

    print("\nQueueOperations")

    lst*=*[]

    size*=*int(input("Enter Size of Queue: "))

    print("Enter Element in Queue")

*for* i *in* range(0,size):

        a*=*int(input())

        lst.append(a)

    print("Element in Queue: ",lst)

    repetQueue(lst)

*#Deque Operations...*

*def* *repetDequeue*(list):

    print("\n\nSelect Deque Operations: ")

    print("\n1]Add Element at Front \n2]Add Element at End \n3]Remove Element at Front\n4]Remove Element at End\n5]Search\n6]Show")

    choiceDeQueue*=*int(input("Enter Choice: "))

*if* choiceDeQueue*==*1:

        addAtFrontDequeue(list)

*elif* choiceDeQueue*==*2:

        addAtLastDequeue(list)

*elif* choiceDeQueue*==*3:

        removeAtFrontDequeue(list)

*elif* choiceDeQueue*==*4:

        removeAtEndDequeue(list)

*elif* choiceDeQueue*==*5:

        serachDequeue(list)

*elif* choiceDeQueue*==*6:

        showDequeue(list)

*else*:

        print("Invalid choice")

        repetQueue(list)

*def* *addAtFrontDequeue*(list):

    colList*=*collections.deque(list)

    num*=*int(input("Enter Element add to Front"))

    colList.appendleft(num)

    list*=*colList

    repetDequeue(list)

*pass*

*def* *addAtLastDequeue*(list):

    colList*=*collections.deque(list)

    num*=*int(input("Enter Element add to Last"))

    colList.append(num)

    list*=*colList

    repetDequeue(list)

*pass*

*def* *removeAtFrontDequeue*(list):

*if* list*==*[]:

        print("Dequeue is empty: ")

*else*:

        colList*=*collections.deque(list)

        colList.popleft()

        list*=*colList

    repetDequeue(list)

*pass*

*def* *removeAtEndDequeue*(list):

*if* list*==*[]:

        print("Dequeue is empty: ")

*else*:

        colList*=*collections.deque(list)

        colList.pop()

        list*=*colList

    repetDequeue(list)

*pass*

*def* *serachDequeue*(list):

    colList*=*collections.deque(list)

    num*=*int(input("\nEnter Number to search: "))

*if* num *in* colList:

        print("Number is present at index: {0}".format(list.index(num)))

*else*:

        print("Number is not present")

    list*=*colList

    repetDequeue(list)

*pass*

*def* *showDequeue*(list):

    print("\nElements in Queue are: ",list)

    repetDequeue(list)

*pass*

*def* *dequeueOp*():

    print("\nDequeue Operations")

    list*=*[]

    size*=*int(input("Enter Size of Dequeue: "))

    print("Enter Element in Dequeue")

*for* i *in* range(0,size):

        a*=*int(input())

        list.append(a)

    print("Element in Dequeue: ",list)

    repetDequeue(list)

*pass*

*def* *repetLL*(lst):

    print("\n\nSelect Stack Operations: ")

    print("\n1]Add Element\n2]Add Element at Given Position\n3]Remove Element\n4]Replace Element\n5]Search\n6]Show")

    choiceLL*=*int(input("Enter Choice: "))

*if* choiceLL*==*1:

        addLL(lst)

*elif* choiceLL*==*2:

        addAnyLL(lst)

*elif* choiceLL*==*3:

        removeLL(lst)

*elif* choiceLL*==*4:

        replaceLL(lst)

*elif* choiceLL*==*5:

        searchLL(lst)

*elif* choiceLL*==*6:

        showLL(lst)

*else*:

        print("Invalid choice")

        repetLL(lst)

*def* *addLL*(lst):

    num*=*int(input("Enter Element add to Linked lst: "))

    lst.append(num)

    repetLL(lst)

*def* *addAnyLL*(lst):

    num1*=*int(input("Enter Index where to add element: "))

    num2*=*int(input("Enter element to add Linked lst: "))

    lst.insert(num1,num2)

    repetLL(lst)

*def* *removeLL*(lst):

    num*=*int(input("Enter Element to remove: "))

    lst.remove(num)

    repetLL(lst)

*def* *replaceLL*(lst):

    num1*=*int(input("Enter Element to Old value "))

    index*=*lst.index(num1)

    num2*=*int(input("Enter Element to New value "))

    lst.insert(index,num2)

    lst.remove(num1)

    repetLL(lst)

*def*  *searchLL*(lst):

    num*=*int(input("\nEnter Number to search: "))

*if* num *in* range(len(lst)):

        print("Number is present at index: {0}".format(lst.index(num)))

*else*:

        print("Number is not present")

    repetLL(lst)

*def* *showLL*(lst):

    print("Linked lst: ",lst)

    repetLL(lst)

*def* *linkedlstOp*():

    print("\nLinked lst Operations")

    lst*=*[]

    size*=*int(input("Enter Linked lst: "))

    print("Enter Element in Linked lst")

*for* i *in* range(0,size):

        a*=*int(input())

        lst.append(a)

    print("Element in Linked lst: ",lst)

    repetLL(lst)

*pass*

print("Select any from following: \n")

print("1]Stack\n2]Queue\n3]Deque\n4]Linked lst\n")

choice *=* int(input("Enter Your choice: "))

*if* choice*==*1:

    stackOp()

*elif* choice*==*2:

    queueOp()

*elif* choice*==*3:

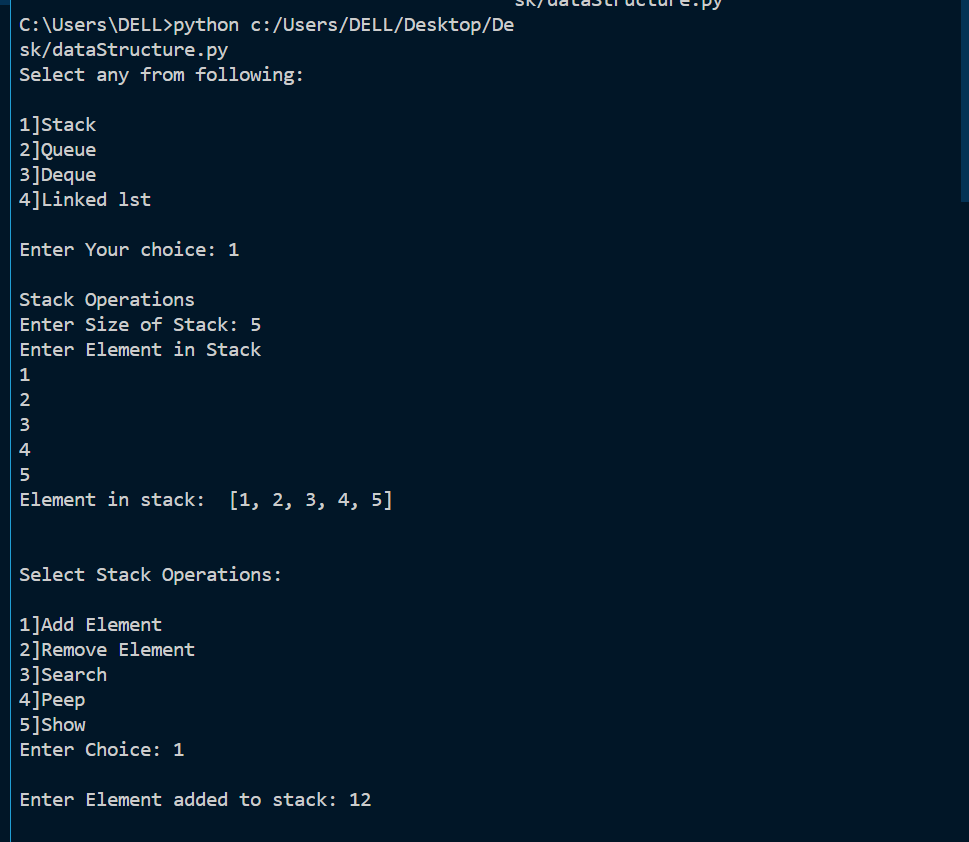
    dequeueOp()

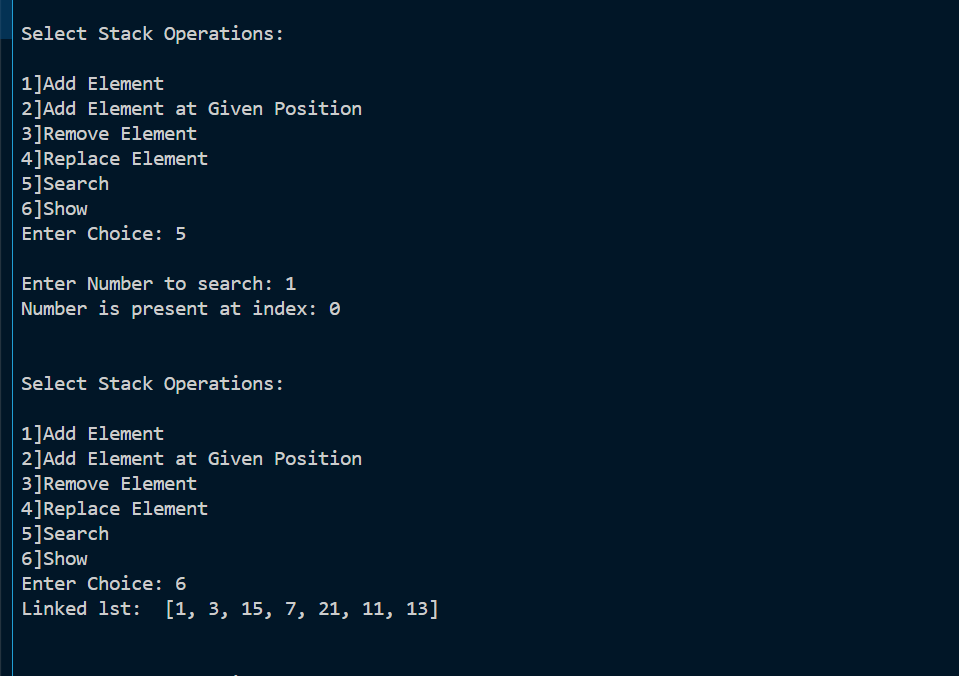
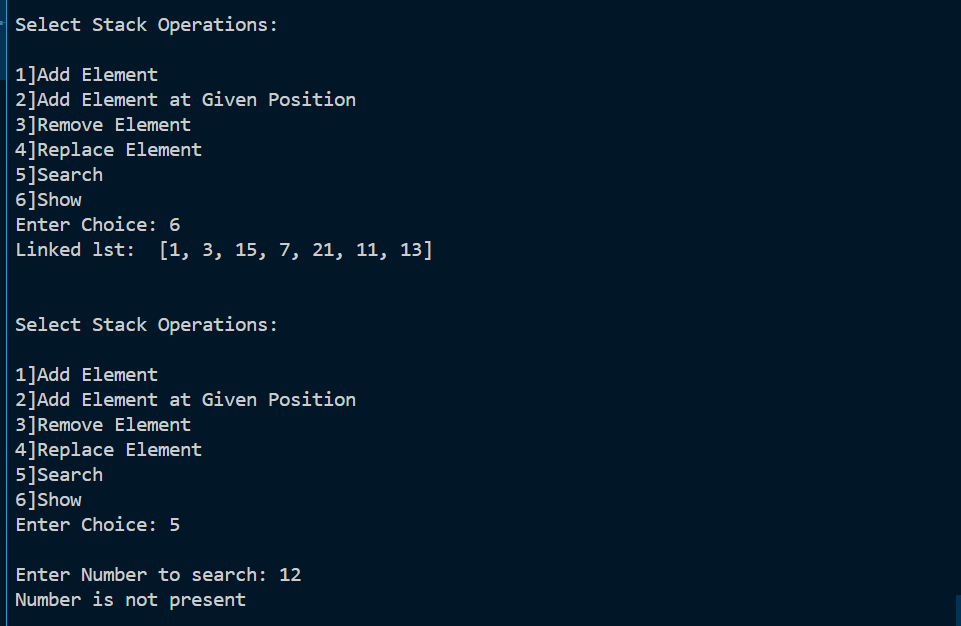
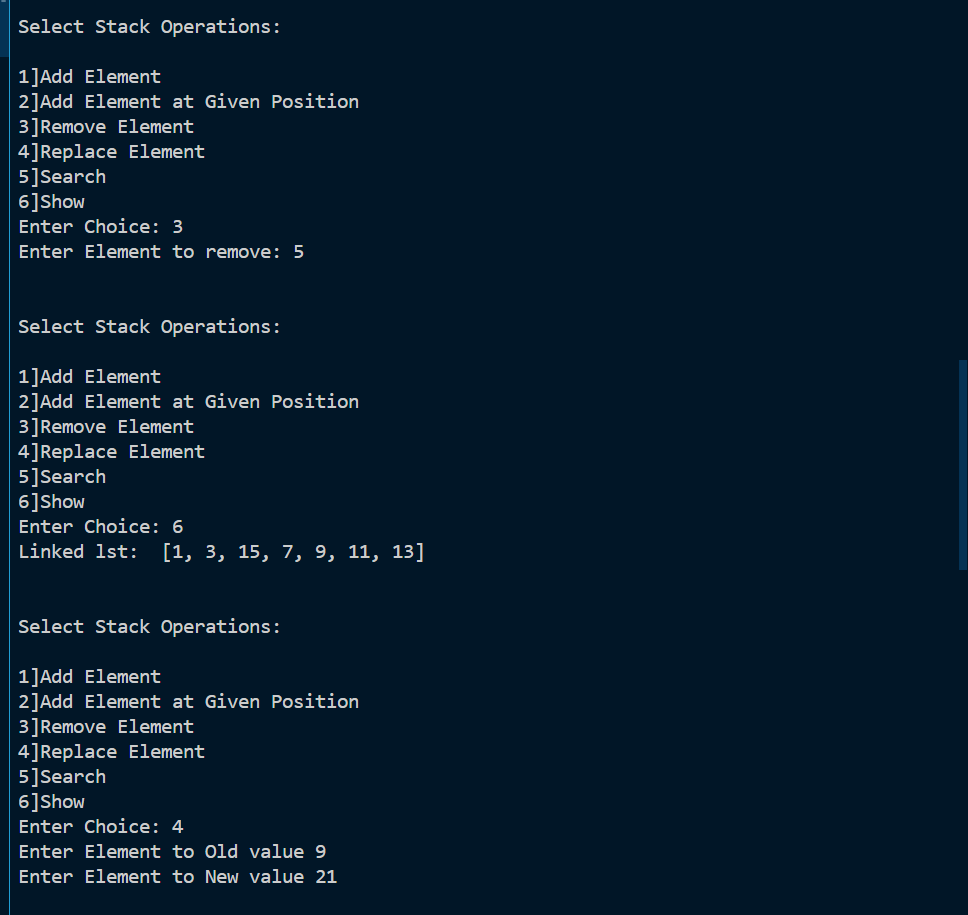
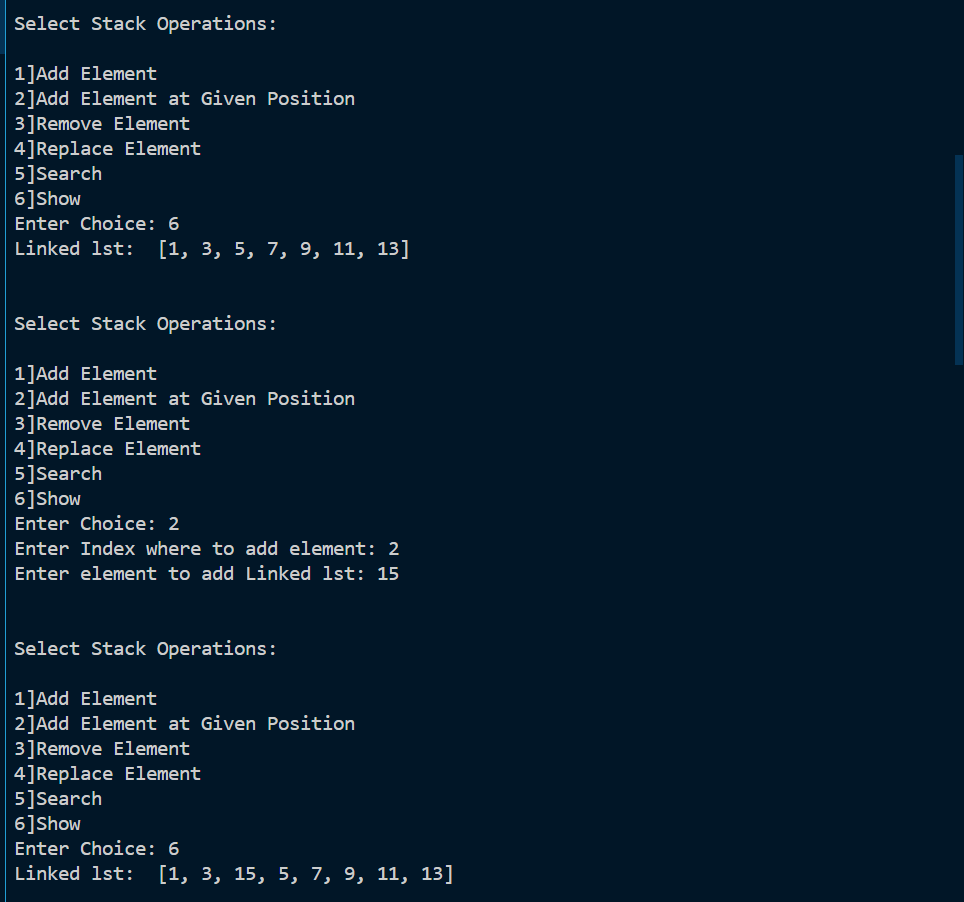
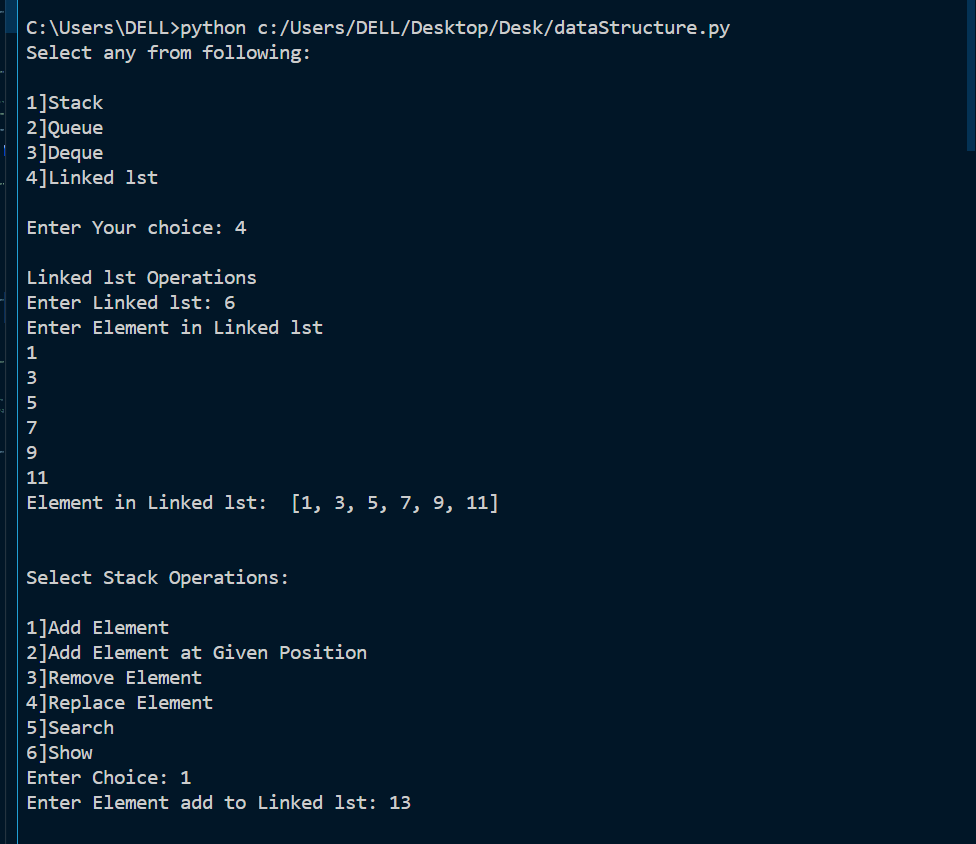
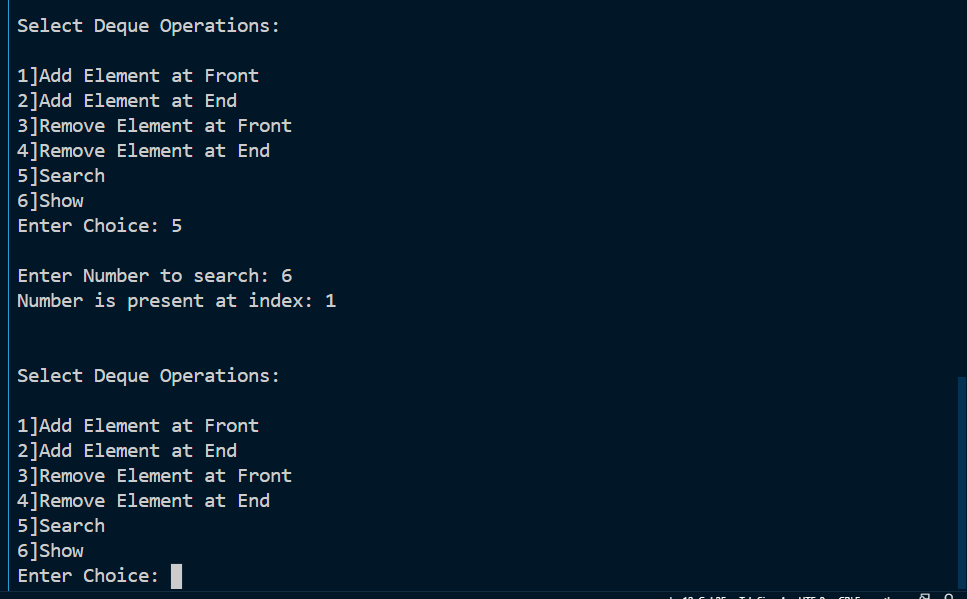
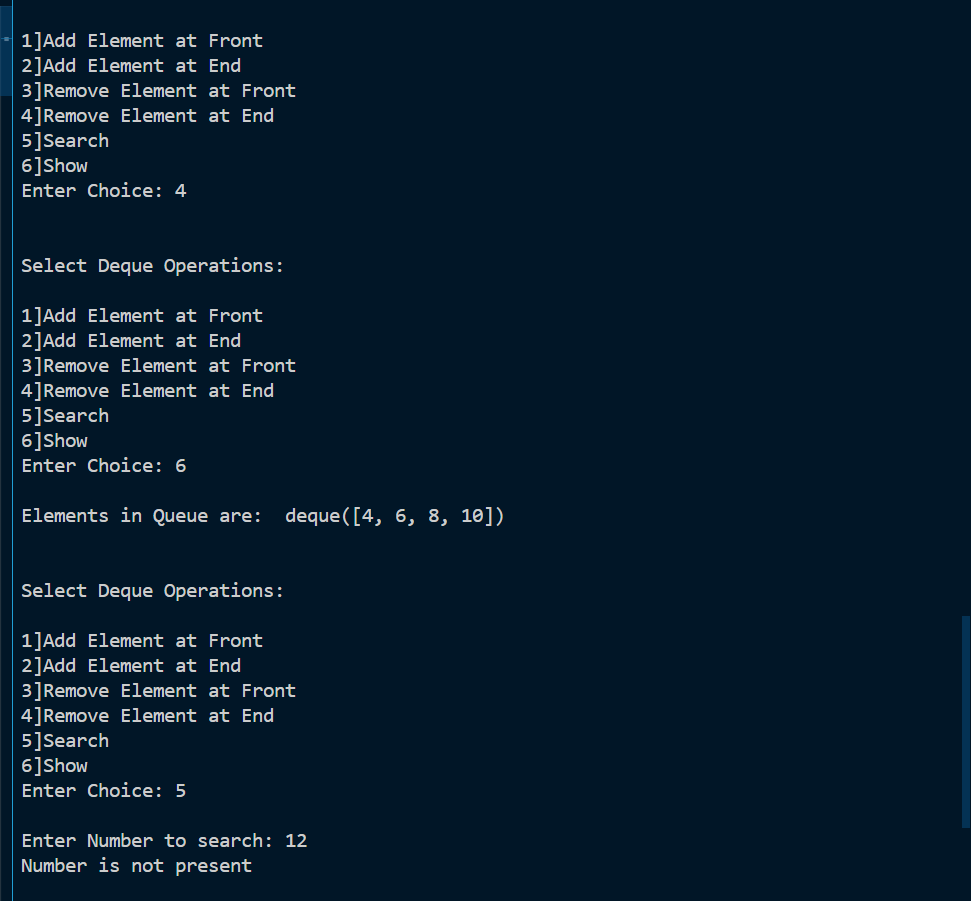
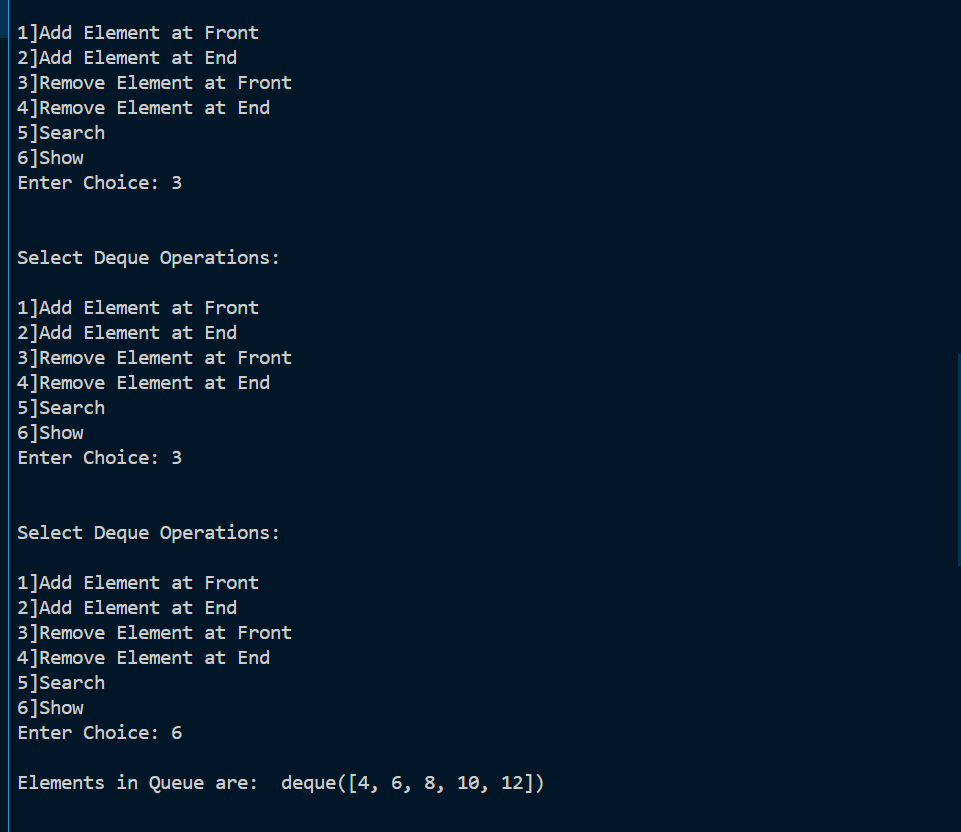
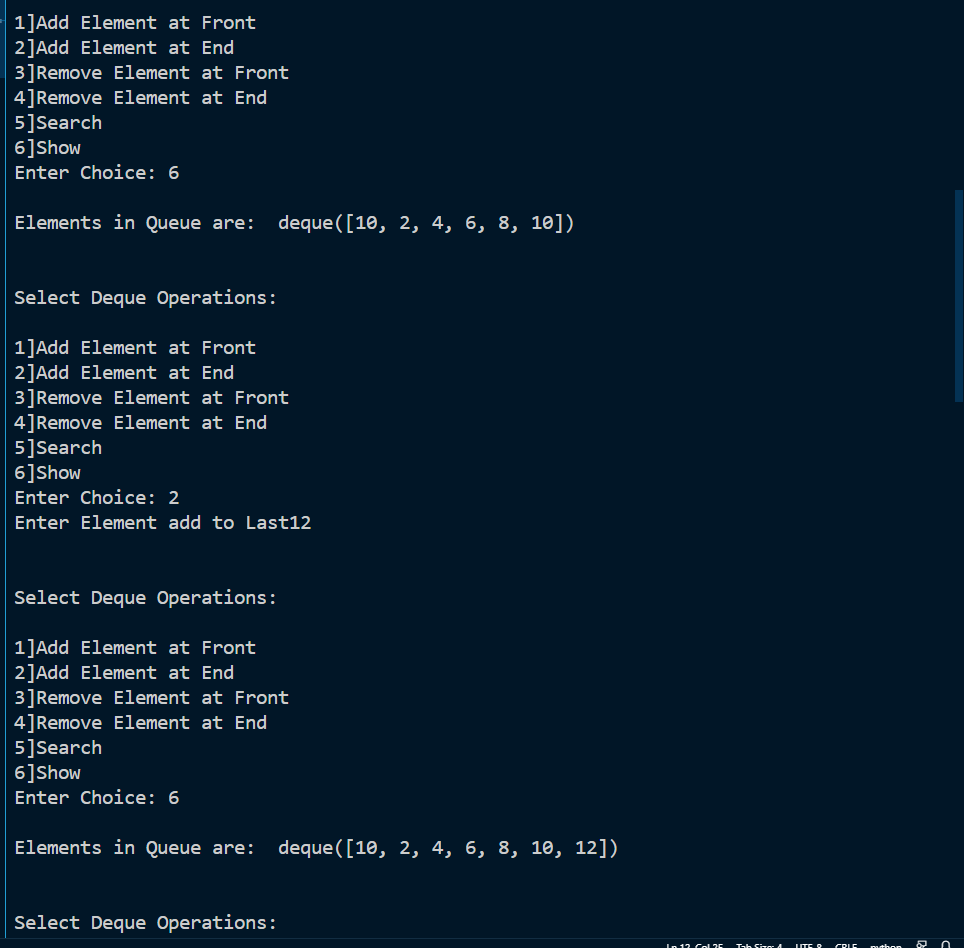
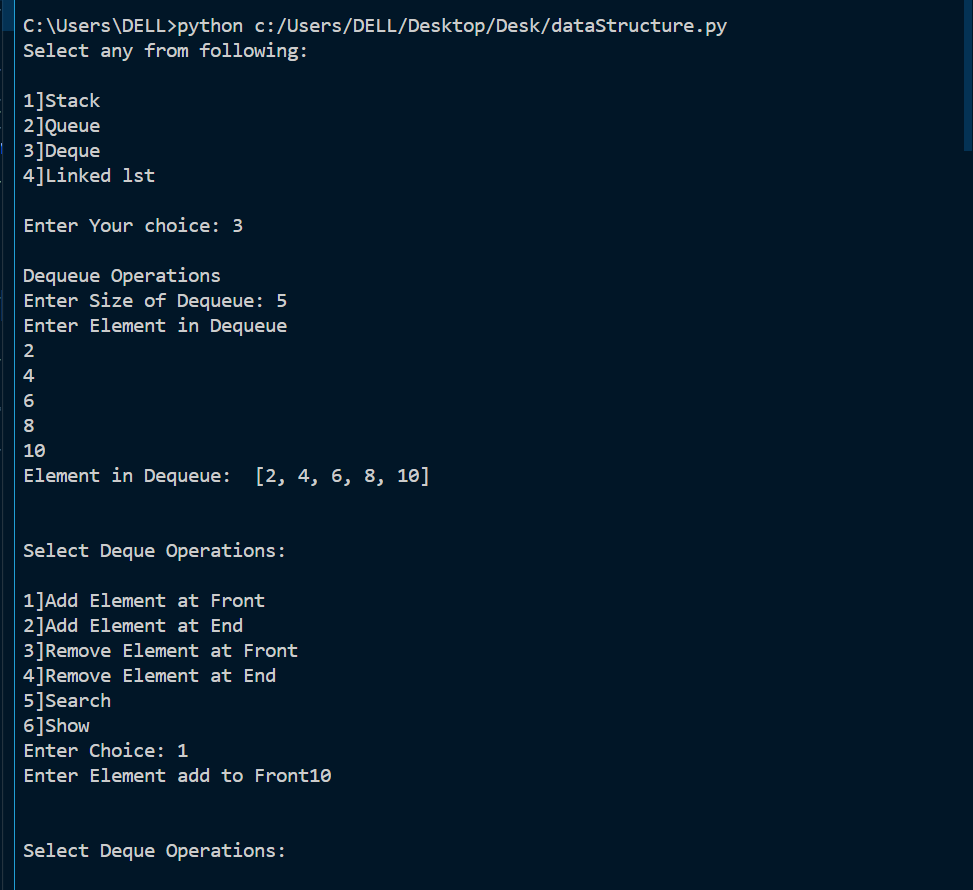
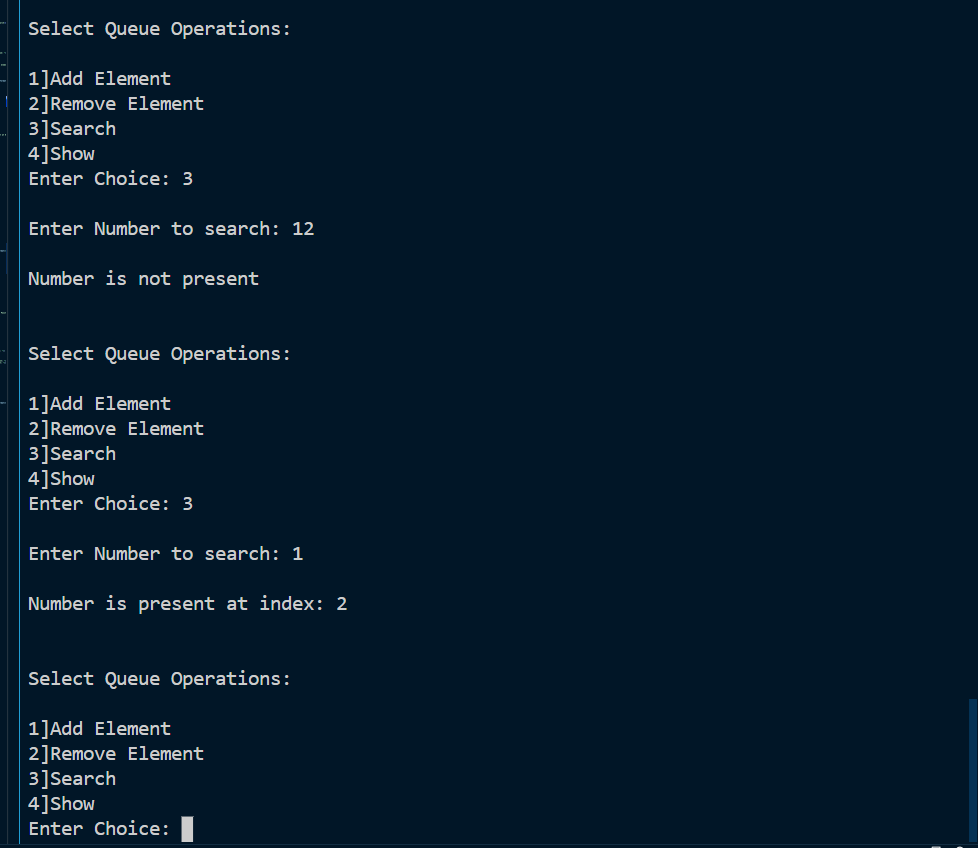
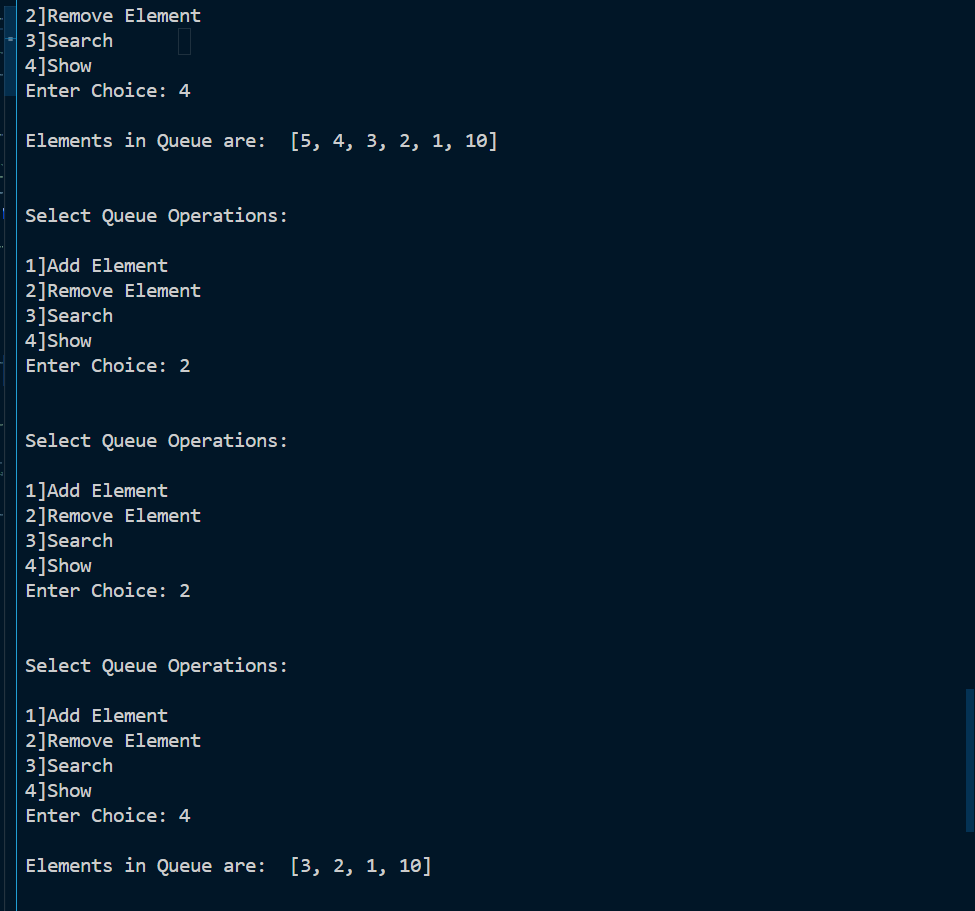
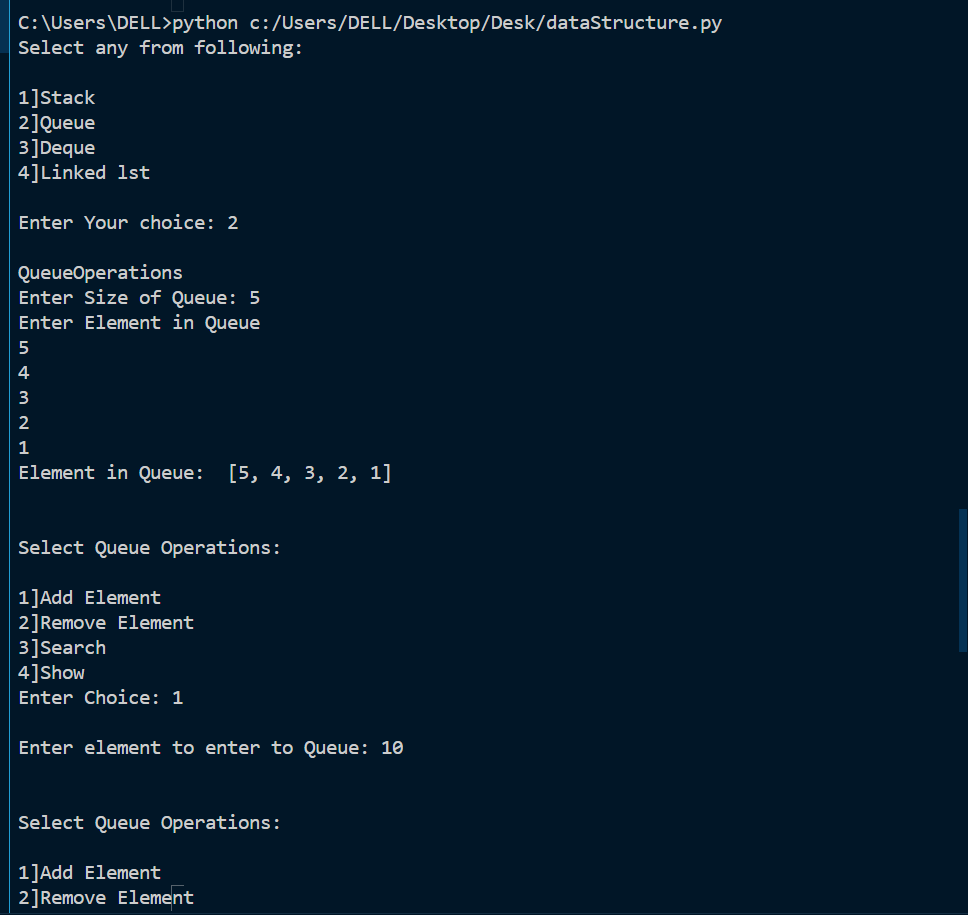
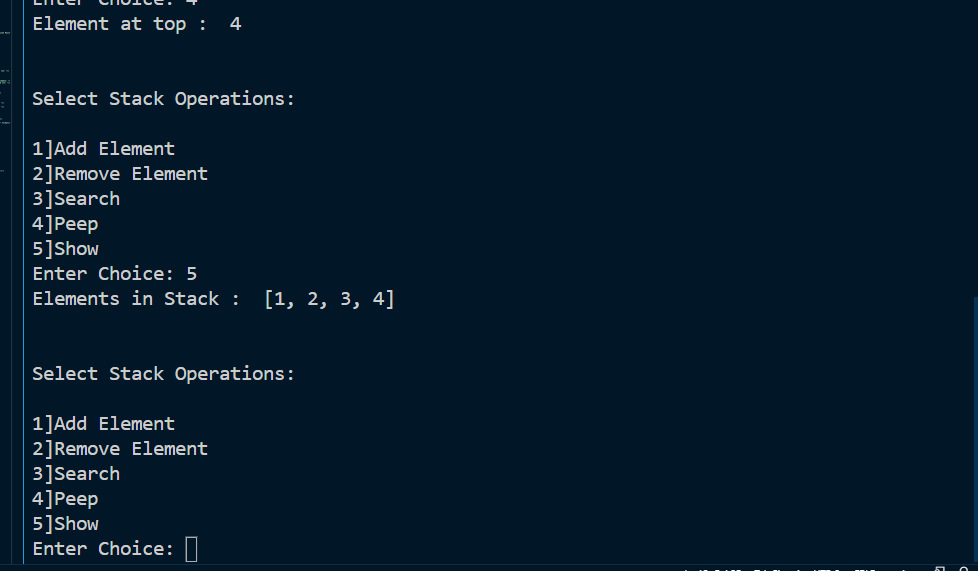
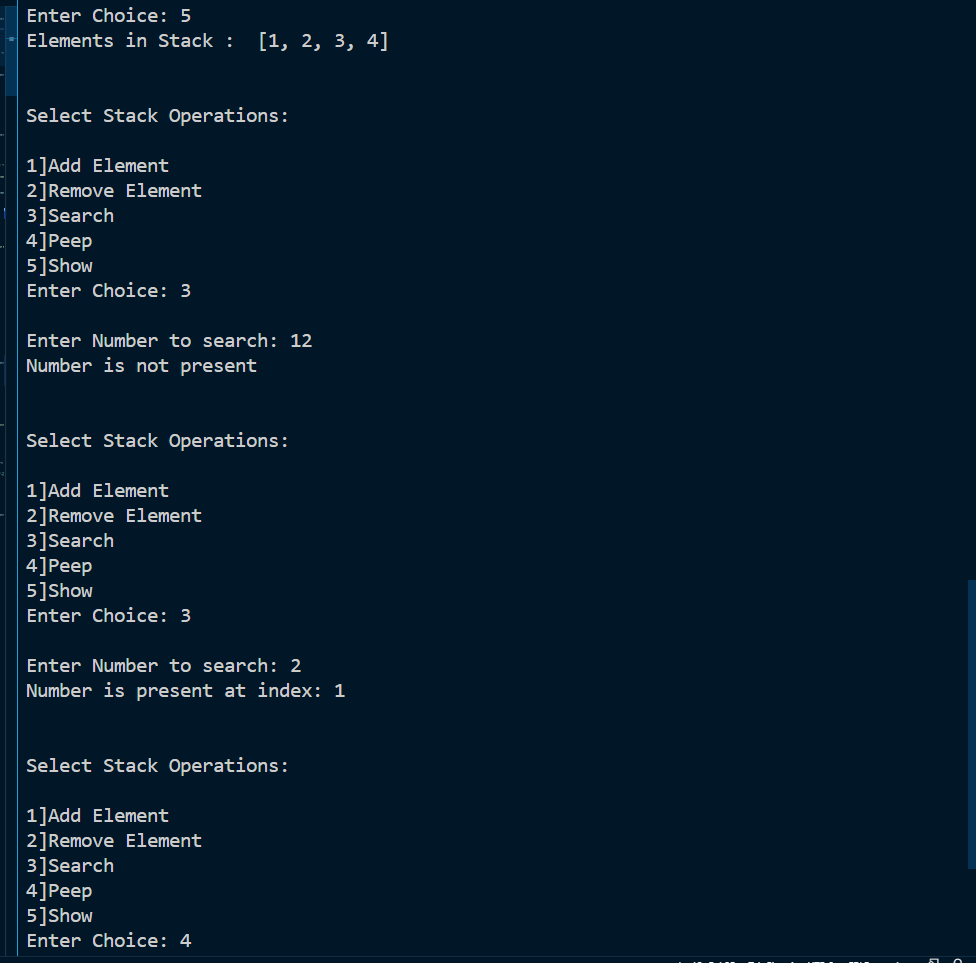
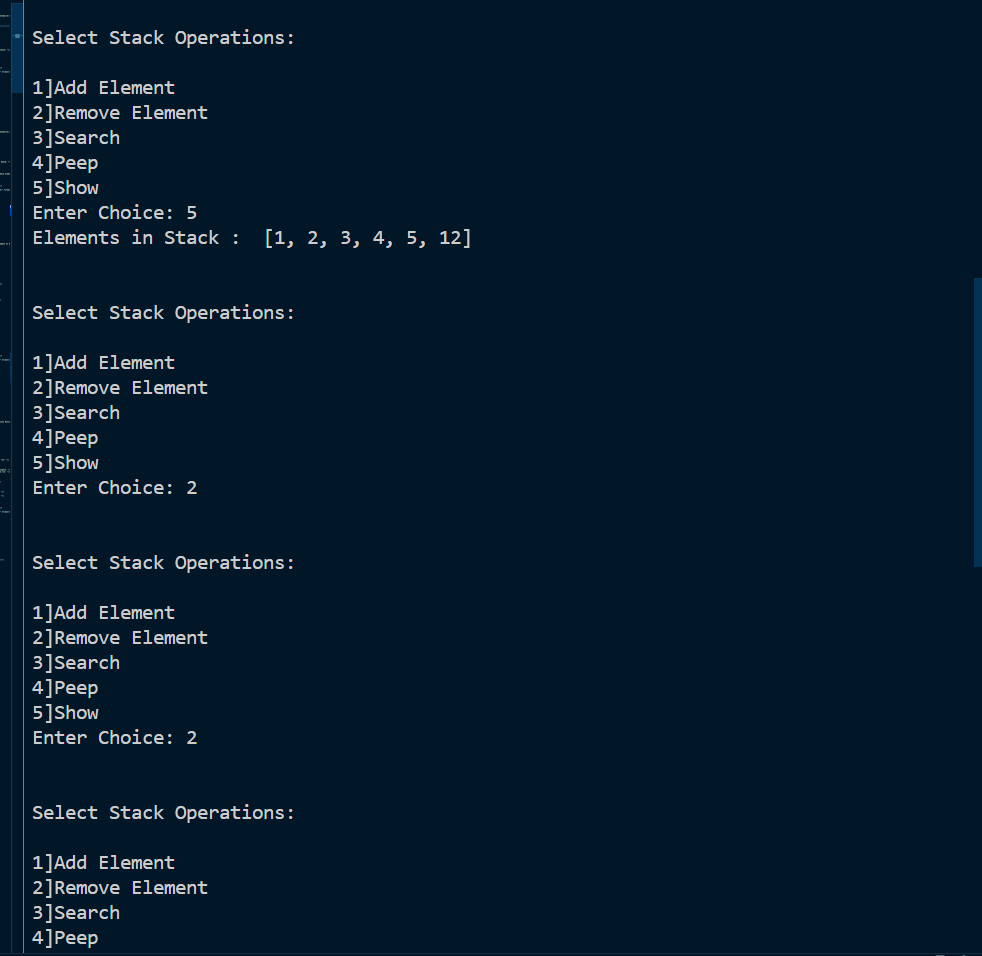
*elif* choice*==*4:

    linkedlstOp()

*else*:

    print("Plz Enter Proper Choice")

**Output:**

****