**PRACTICAL 1**

**Demonstration of stack using list in python**

**Demonstration of Stack:**

**1] Insertion:**

**Code:**

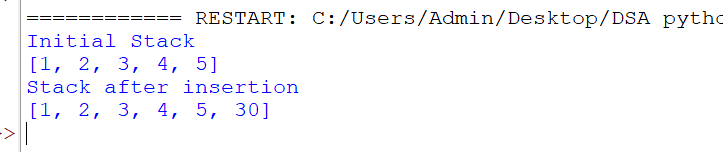
stack=[1,2,3,4,5]

print("Initial Stack")

print(stack)

stack.append(30)

print("Stack after insertion")

print(stack)

**2] deletion**

stack=[1,2,3,4,5,6,7,8]

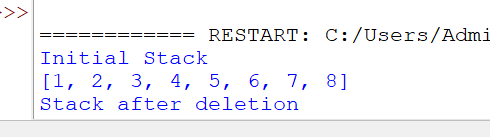
print("Initial Stack")

print(stack)

stack.pop()

print("Stack after deletion")

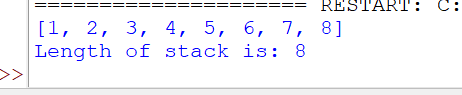
print(stack)



**3] Top Element**

stack=[1,2,3,4,5,6,7,8]

print(stack)

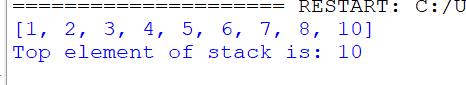
print("Length of stack is:" ,(len(stack)))

**4] Element at the top:**

**Code:**

stack=[1,2,3,4,5,6,7,8,10]

print(stack)

print("Top element of stack is:" ,(stack[-1]))

**Practical 2**

**Demonstration of Queue:**

**1] Insertion:**

**Code:**

queue=[1,2,3,4,5]

print("Initial Queue")

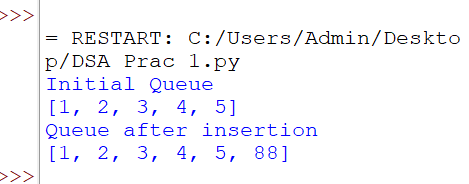
print(queue)

queue.append(88)

print("Queue after insertion")

print(queue)

**Output:**



**2] Deletion:**

**Code:**

queue=[1,2,3,4,5]

print("Initial Queue")

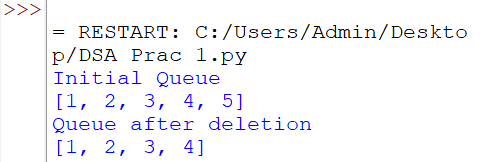
print(queue)

queue.pop()

print("Queue after deletion")

print(queue)

**Output:**



**3] Size:**

**Code:**

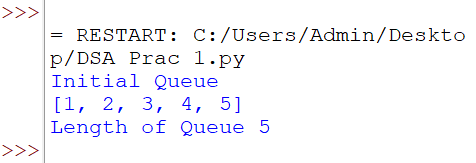
queue=[1,2,3,4,5]

print("Initial Queue")

print(queue)

print("Length of Queue",len(queue))

**Output:**

****

**4] Element at the top:**

**Code:**

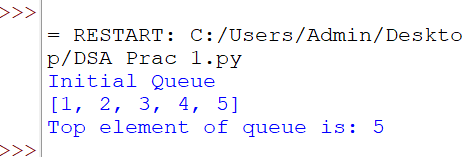
queue=[1,2,3,4,5]

print("Initial Queue")

print(queue)

print("Top element of queue is:" ,(queue[-1]))

**Output:**

****

**B] Using deque:**

**1] Insertion:**

**Code:**

a=deque();

a.append('P');

a.append('R');

a.append('A');

a.append('T');

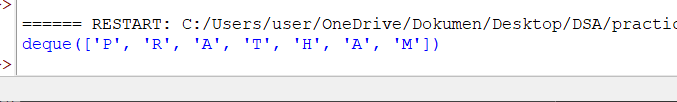
a.append('H');

a.append('A');

a.append('M');

print(a);

**Output:**

****

**2] Deletion:**

**Code:**

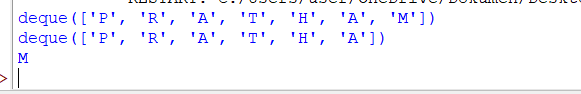
**partA**

b=a.pop();

print(a);

print(b);

**Output:**

****

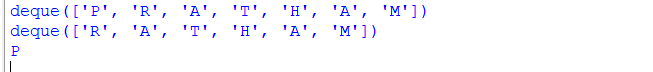
partB

b=a.popleft()

print(a);

print(b);

**output:-**

****

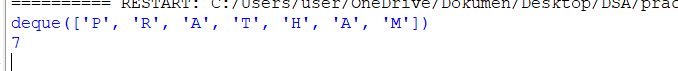
**3] Size:**

**Code:**

l=len(a)

print(l);

**Output:**

****

**Practical 3**

1. **Linear search**

a=[1,2,3,4,5,6,7]

b=int(input("enter a number:"))

for i in a:

if i==b:

c=1

break

else:

c=0

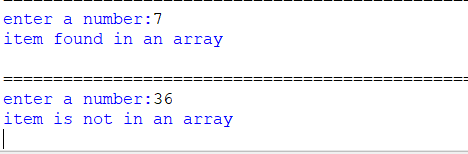
if c==1:

print("item found in an array")

else:

print("item is not in an array")

**output**



**Practical 4**

**Recursion**

1. **Factorial**

a=int(input("enter a number:"))

def factorial(n):

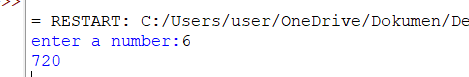
if n==1:

return 1

else:

return n\* factorial(n-1)

print(factorial(a))

**output:-**

**2)Fibonacci series**

a=int(input("enter a number:"))

def fibonaaci(n):

if n==0:

return 0

elif n==1:

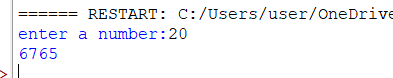
return 1

else:

return fibonaaci(n-1)+fibonaaci(n-2)

print(fibonaaci(a))

**output:-**

****

**Practical 5**

**Binary Search**

def binary\_search(arr,x):

LB=0

UB=len(arr)-1

MID=0

while LB<=UB:

MID=(UB+LB)

if arr[MID]<x:

LB=MID+1

elif arr[MID]>x:

UB=MID-1

else:

return MID

return-1

arr=[10,20,30,40,50,60,70]

x=int(input("enter number :"))

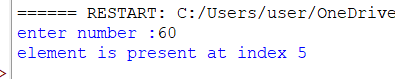
result=binary\_search(arr,x)

if result!=-1:

print("element is present at index", str(result))

else:

print("element is not present in arrya")

**output**