**STACK IMPLEMENTATION:**

class stack:

    def \_\_init\_\_(self):

        self.l=[]

    def push(self,ele):

        self.l.append(ele)

    def peek(self):

        return self.l[-1]

    def size(self):

        return len(self.l)

    def is\_empty(self):

        if(len(self.l)==0):

            return True

        return False

    def pop(self):

        if(self.is\_empty()):

            return "The Stack is Empty..."

        return self.l.pop()

    def \_\_str\_\_(self):

        return str(self.l)

s=stack()

s.push(10)

s.push(20)

s.push(30)

print(s)

print(s.peek())

s.pop()

s.pop()

print(s)

s.push(100)

print(s)

print(s.size())

print(s.pop())

print(s.pop())

print(s.pop())

print(s)

# BY IMORTING MODULE DEQUE

from collections import deque

stack=deque()

stack.append(10)

stack.append(20)

stack.append(30)

stack.append(40)

stack.append(50)

print(stack)

stack.pop()

print(stack)

stack.append(100)

print(stack)

**QUEUE IMPLEMENTATION:**

class queue:

    def \_\_init\_\_(self):

        self.l=[]

    def enque(self,ele):

        self.l.append(ele)

    def size(self):

        return len(self.l)

    def is\_empty(self):

        if(len(self.l)==0):

            return True

        return False

    def pop(self):

        if(self.is\_empty()):

            return "QUEUE IS EMPTY..."

        return self.l.pop(0)

    def \_\_str\_\_(self):

        return str(self.l)

q=queue()

q.enque(10)

q.enque(20)

q.enque(30)

q.enque(40)

print(q)

q.pop()

q.pop()

q.pop()

print(q.size())

print(q)

q.enque(100)

print(q)

q.pop()

print(q.pop())

print(q.pop())

print(q)

# BY IMORTING MODULE DEQUE

from collections import deque

q=deque()

q.append(10)

q.append(20)

q.append(30)

q.append(40)

print(q)

q.popleft()

print(q.popleft())

print(q)

**BINARY SEARCH:**

array=[]

for i in range(5):

    n=input("enter a number:")

    array.append(n)

s=0;e=4

num=input("enter number to search:")

while(s<=e):

    mid=(s+e)//2

    if(array[mid]==num):

         print("FOUND")

         exit()

    elif(num<array[mid]):

        e=mid-1

    elif(num>array[mid]):

        s=mid+1

print("NOT FOUND")

# BY RECURSIVE FUNCTION

def recur(s,e,arr,ele):

    if(s>e):

        return 0

    mid=(s+e)//2

    if(arr[mid]==ele):

        return 1

    if(arr[mid]>ele):

        return recur(s,mid-1,arr,ele)

    else:

        return recur(mid+1,e,arr,ele)

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

if((recur(0,4,arr,100))==1):

    print("FOUND")

else:

    print("NOT FOUND")