Assignment 6

Student Name: Surbhi Priya **UID: 22BCS10268**
Branch: CSE **Section/Group: 22BCS_IOT_612**
Semester: 6th **Date of Performance:17/03/25**
Subject Name: Advance prog. Lab **Subject Code: 22CSP-351**

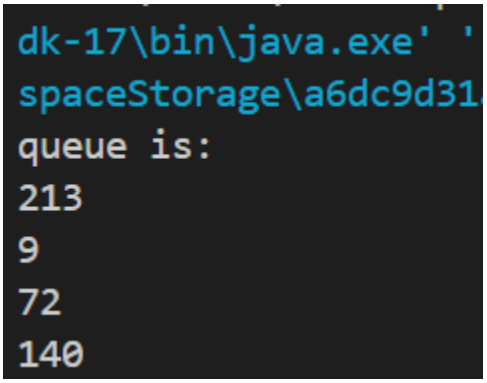
Q1)Implement Queue using Stack

- **Code:**

```
import java.util.*;

public class queueUsingStacks {
   static class queue{
      static Stack<Integer> s1=new Stack<>();
      static Stack<Integer> s2=new Stack<>();

      public static boolean isEmpty(){
         return s1.isEmpty();
      }
      public static void add(int data){
         while(!s1.isEmpty()){
            s2.add(s1.pop());
         }
         s2.push(data);
         while(!s2.isEmpty()){
            s1.add(s2.pop());
         }
      }
      public static int remove(){
         if(s1.isEmpty() && s2.isEmpty()){
            System.out.println("Queue is empty");
            return -1;
         }
         return s1.pop();
      }
      public static int peek(){
         if(s1.isEmpty() && s2.isEmpty()){
            System.out.println("Queue is empty");
            return -1;
         }
         return s1.peek();
      }
   }
   public static void main(String[] args) {
      queue q=new queue();
      q.add(213);
```

```
            q.add(9);
            q.add(72);
            q.add(140);
            while(!q.isEmpty()){
                System.out.println(q.peek());
                q.remove();
            }
        }
    }
```

- **Screenshot:**



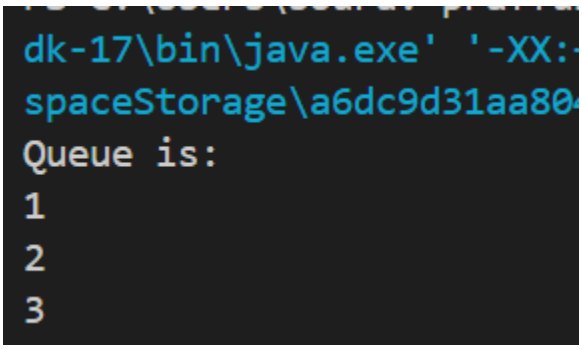
Q2) <u>Implement Queue Using an Array</u>

- **Code:**

```
public class QueueImplUsingArrays {
    static class queue{
        static int[]arr;
        static int size;
        static int rear;
        queue(int n){
            arr=new int[n];
            this.size=n;
            rear = -1;
        }
        //to check isEmpty
        public static boolean isEmpty(){
            return rear==-1;
        }
        //to add element
        public static void add(int el){
            if(rear==size-1){
                System.out.println("queue is already full");
                return;
```

```
            }
            rear++;
            arr[rear]=el;
        }
        //dequeue
        public static int remove(){
            if(isEmpty()){
                System.out.println("Empty queue");
                return -1;
            }
            int front=arr[0];
            for(int i=0;i<rear;i++){
                arr[i]=arr[i+1];
            }
            rear--;
            return front;
        }
        //front
        public static int peek(){
            if(isEmpty()){
                System.out.println("Empty queue");
                return -1;
            }

            return arr[0];
        }
    }
    public static void main(String[] args) {
        queue q=new queue(5);
        System.out.println("Queue is:");
        q.add(1);
        q.add(2);
        q.add(3);
        while(!q.isEmpty()){
            System.out.println(q.peek());
            q.remove();
        }

    }
}
```

- **Screenshot:**



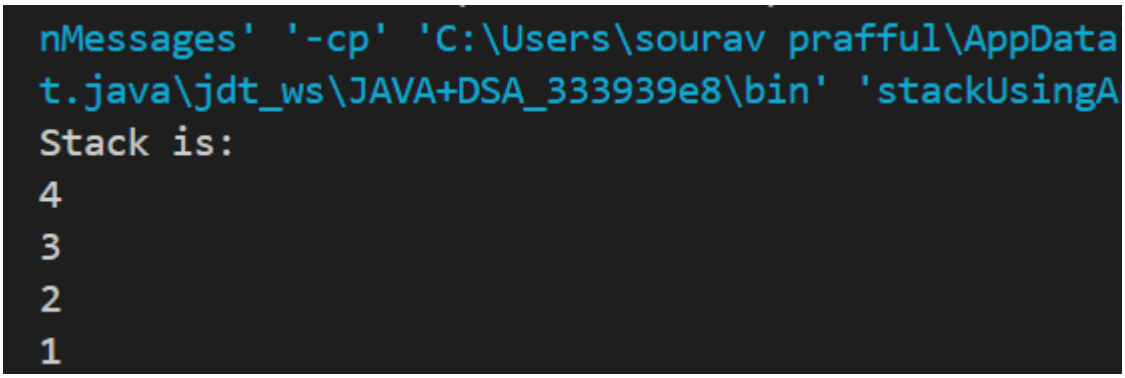
Q3) Stack using ArrayList

- **Code:**

```
import java.util.ArrayList;
public class stackUsingArraylist {
    static class Stack {
        ArrayList<Integer> list = new ArrayList<>();

        public void push(int data) {
            list.add(data);
        }

        public boolean isEmpty() {
            return list.size() == 0;
        }

        public int pop() {
            if(isEmpty()) {
                return -1;
            }
            int top = list.remove(list.size()-1);
            return top;
        }

        public int peek() {
            if(isEmpty()) {
                return -1;
            }
            return list.get(list.size()-1);
        }
    }
    public static void main(String args[]) {
        Stack stack = new Stack();
        stack.push(1);
        stack.push(2);
        stack.push(3);
```

```
        stack.push(4);

        while(!stack.isEmpty()) {
            System.out.println(stack.peek());
            stack.pop();
        }
    }
}
```

- **Screenshot:**



Q4) Implement Stack using LinkedList

- **Code:**

```
public class stackUsingLinkedlist {
    public static class Node{
        int data;
        Node next;
        Node(int data){
            this.data=data;
            next=null;
        }
    }
    static class stack{
        public static Node head=null;
        public static boolean isEmpty(){
            return head==null;
        }
        public static void push(int data){
            Node newNode=new Node(data);
            if(head==null){
                head=newNode;
                return;
            }
            newNode.next=head;
            head=newNode;
        }
        public static int pop(){
```

```
            if(isEmpty()){
               return -1;
            }
            int top=head.data;
            head=head.next;
            return top;
         }
         public static int peek(){
            if(isEmpty()){
               return -1;
            }
            return head.data;
         }
      }
      public static void main(String[] args) {
         stack stack = new stack();
         stack.push(1);
         stack.push(2);
         stack.push(3);
         stack.push(4);

         while(!stack.isEmpty()) {
            System.out.println(stack.peek());
            stack.pop();
         }

      }

   }
```

- **Screenshot:**

```
Stack Implementation using linkedList:
4
3
2
1
```

Q5) Queue Implementation using LinkedList

- **Code:**

```
public class queueLLimpl {
   static class Node{
      int data;
      Node next;
      Node(int data){
         this.data=data;
         next=null;
```

```
    }
}
static class queue{
    static Node head=null;
    static Node tail=null;
    //isEmpty
    public static boolean isEmpty(){
        return head==null && tail==null;
    }
    //no need to check isFull because linked list is of variable size
    public static void add(int data){
        Node newNode=new Node(data);
        if(tail==null){
            tail=head=newNode;
        }
        tail.next=newNode;
        tail=newNode;
    }
    public static int remove(){
        if(isEmpty()){
            System.out.println("linkedList is empty");
            return -1;
        }
        int front=head.data;
        if(head==tail){
            tail=null;
        }
        head=head.next;
        return front;
    }
    public static int peek(){
        if(isEmpty()){
            System.out.println("linkedList is empty");
            return -1;
        }
        return head.data;
    }

}
public static void main(String[] args) {
    queue q=new queue();
    q.add(23);
    q.add(9);
    q.add(7);
    q.add(10);
    while(!q.isEmpty()){
        System.out.println(q.peek());
        q.remove();
    }
```

```
    }
}
```

- **Screenshot:**