**DATA STRUCTURE ASSESSMENT:**

**ANSWERS:**

1. **Deleting a node whose location is given.**
2. **Rear node.**
3. **The function is implemented incorrectly as it changes node.**
4. **Operation on stack are performed smoothly.**
5. **2.**
6. **B.**
7. **O(1)and o(n)**
8. **12a-b9+\*d4\*/**
9. **Print all the nodes in the linked list but in resverse order**
10. **Ollehdlrow**

**CODING:**

1. import java.util.Scanner;

class LinkedList

{

 Node head;

  static class Node  {

 int data;

 Node next;

 Node(int d)

 {

 data = d;

 next = null;

 }

 }

 void deleteNode(

Node temp = head,

prev = null;

 if (temp != null && temp.data == key)

 {

 head = temp.next; // Changed head

 return;

 }

 while (temp != null && temp.data != key)

 {

 prev = temp;

 temp = temp.next;

 }

 if (temp == null) {

 System.out.println("The node of value "+ key +" is not available");

 return;

 }

 prev.next = temp.next;

 }

 public void push(int new\_data)

 {

 Node new\_node = new Node(new\_data);

 new\_node.next = head;

 head = new\_node;

 }

public void printList()

 {

 Node tnode = head;

 while (tnode != null)

 {

 System.out.print(tnode.data+" ");

 tnode = tnode.next;

 }

 System.out.println();

 }

   public static void main(String[] args)

 {

 LinkedList llist = new LinkedList();

 Scanner sc = new Scanner(System.in);

 int n=sc.nextInt();

 for(int i=0;i<n;i++)

 {

 int x=sc.nextInt();

 llist.push(x);

 }

 int key;

 key = sc.nextInt();

 llist.printList();

/

llist deleteNode(key);

 llist.deleteNode(key);  llist.printList();

 }

}

2) import java.util.Scanner;

import java.util.Stack;

class Test {

 static Stack<Integer> st = new Stack<>();

 public static void main(String[] args)

 {

 Scanner sc=new Scanner(System.in);

 int n=sc.nextInt();

 for(int i=0;i<n;i++)

 {

 int x=sc.nextInt();

 st.push(x);

 }

 System.out.println(st);

 reverse();

 System.out.println(st);

 }

 static void reverse(){

 if(st.size() > 0)

 {

 int x = st.peek();

 st.pop();

 reverse();

 insert\_at\_bottom(x);

 }

 }

 static void insert\_at\_bottom(int x)

 {

 if(st.isEmpty())

 st.push(x);

 else

 {

 st.pop();

 insert\_at\_bottom(x);

 st.push(a);

 }