**Java Lab 7**

**Done By:** Rohit Karunakaran **Roll No:** 58

**Date**: 30-10-2020

**Experiment -**

**Aim:** Write a Java program for the following:

1. Create a doubly linked list of elements.
2. Delete a given element from the above list.
3. Display the contents of the list after deletion.

**Concepts Used:** Doubly Linked List

**Algorithm:**

Class Node

Data Members

data:int

prev : Node

next :Node

Class DoublyLinkedList

Data Members

Header: Node

size: int

Methods()

addNode(): void

removeNode(): void

display(): void

**Algorithm addNode( int elem):**

Start

Node n = new Node()

if n!=NULL then

n->data = elem

Node n2 = Header

while(n2->next!=NULL) do

n2 = n2->next

endWhile

n2->next = n

n->next = NULL

n->prev = n2

endif

Stop

**Algorithm removeNode(int a)**

Start

if Header->next!=NULL then

flag=0

Node ptr = Header->next

while ptr -> next!=NULL do

if ptr->data == a then

flage =1

endif

endwhile

if flag == 1 then

if(ptr->next!=NULL) then

ptr->next->prev = ptr->prev

endif

ptr->prev->next = ptr->next

ptr->prev=NULL

ptr->next = NULL

else

print “element not found”

endif

endif

Stop

**Program Code:**

/\*Doubly Linked List implementation in Java

\* Done By: Rohit Karunakaran

\* \*/

import java.io.\*;

/\*Node class for the nodes of the linked list\*/

class Node

{

private int data;

private Node prev;

private Node next;

//Constructors

public Node(int x,Node next,Node prev)

{

data = x;

this.prev =prev;

this.next = next;

}

public Node(int x){ this(x,null,null); }

public Node() { this(0); }

//Gettes and Setters

public Node getNextNode() { return this.next; }

public Node getPrevNode() { return this.prev; }

public int getData() { return this.data; }

public void setNextNode(Node n) { this.next=n; }

public void setPrevNode(Node n) { this.prev=n; }

public void setNextNode() { this.next=null; }

public void setPrevNode() { this.prev=null; }

}

/\* Doubly linked list class that contains the relevent functions for

\* implementation\*/

class DoublyLinkedList

{

private Node header; //The header node

public int length; //To keep a track of the length of the doubly linked list

public DoublyLinkedList()

{

header = new Node();

length=0;

}

public DoublyLinkedList(int nums[]) //Creates a doubly linked list when an array of numbers is passed

{

this();

for(int x:nums)

{

this.add(x);

}

}

public void add(int x) //add a node to the end of the doubly linked list

{

Node ptr=header;

while(ptr.getNextNode()!=null)

ptr=ptr.getNextNode();

Node n = new Node(x);

n.setPrevNode(ptr);

n.setNextNode(ptr.getNextNode());

ptr.setNextNode(n);

length++;

}

public void remove(int x) //remove the node containing the given value if it exists

{

Node ptr=header.getNextNode();

if(ptr==null)

{

System.out.println("The List is empty");

return;

}

while(ptr!=null)

{

if(ptr.getData() == x)

break;

ptr=ptr.getNextNode();

}

if(ptr!=null)

{

//delete node

if(ptr.getNextNode()!=null)

ptr.getNextNode().setPrevNode(ptr.getPrevNode());

ptr.getPrevNode().setNextNode(ptr.getNextNode());

ptr.setNextNode(null);

ptr.setPrevNode(null);

length--;

}

else

{

System.out.println("No Such element found");

}

}

public void displayList()

{

Node ptr=header.getNextNode();

while(ptr!=null)

{

System.out.println(ptr.getData());

ptr=ptr.getNextNode();

}

}

}

public class MainClass

{

public static void main(String args[]) throws IOException

{

DoublyLinkedList dll = new DoublyLinkedList();

int elem=0;

BufferedReader r = new BufferedReader(new InputStreamReader(System.in));

boolean Run = true;

while(Run)

{

System.out.println("\n-------------Menu------------");

System.out.println("1.Add an element");

System.out.println("2.Remove an element");

System.out.println("3.Display the List");

System.out.println("4.Exit");

System.out.print("\nEnter your choice: ");

try

{

int c = Integer.parseInt(r.readLine());

switch(c)

{

case 1: //add an element

System.out.print("Enter the elemet to be added: ");

elem = Integer.parseInt(r.readLine());

dll.add(elem);

break;

case 2: //remove an element

System.out.print("Enter the elemet to be deleted: ");

elem = Integer.parseInt(r.readLine());

dll.remove(elem);

break;

case 3: //display the list

System.out.println("\nThe List is :");

dll.displayList();

break;

case 4:Run = false;

break;

default: System.out.println("Please Enter a valid input ");

break;

}

}

catch (NumberFormatException e)

{

System.out.println("Please Enter a integer value ");

e.printStackTrace();

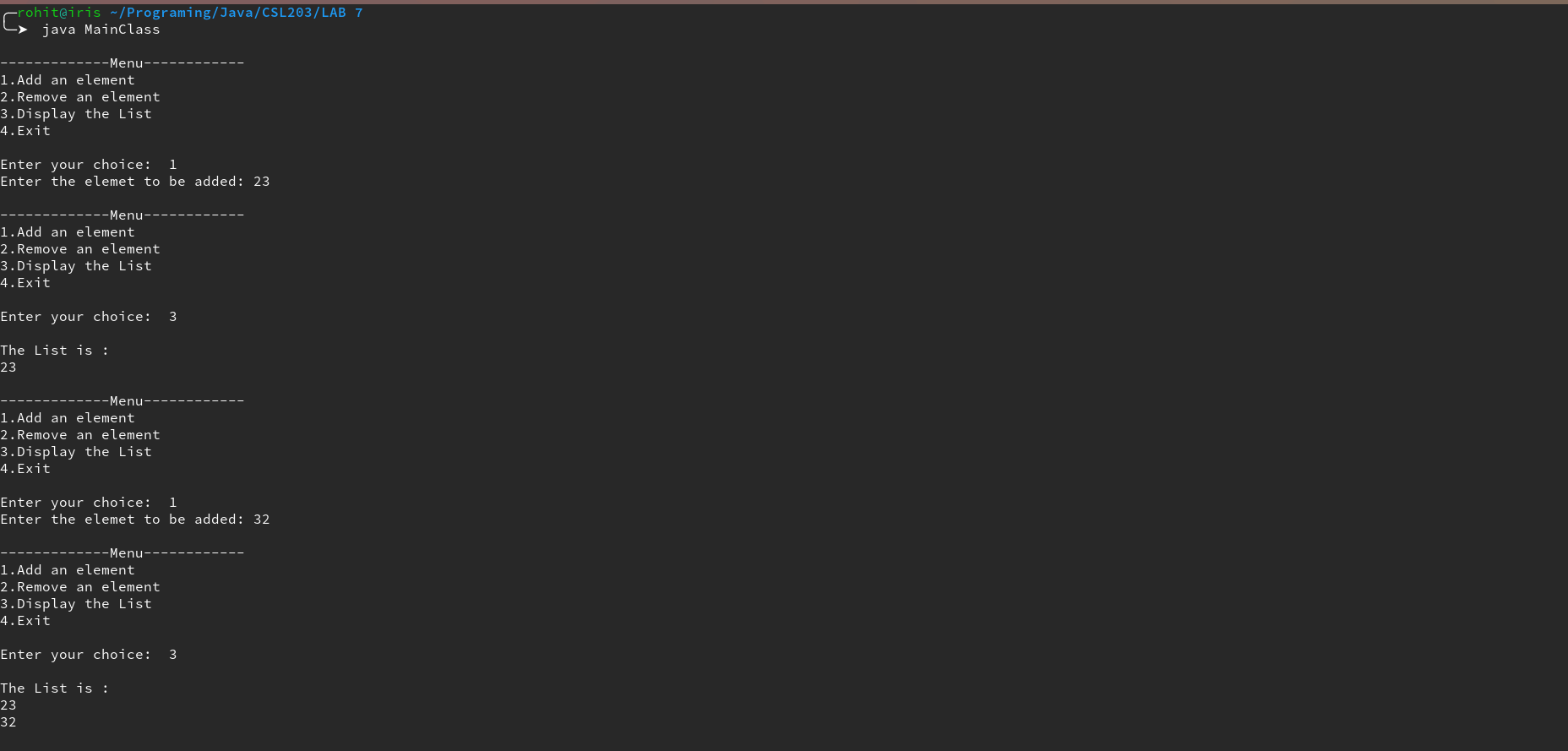
}

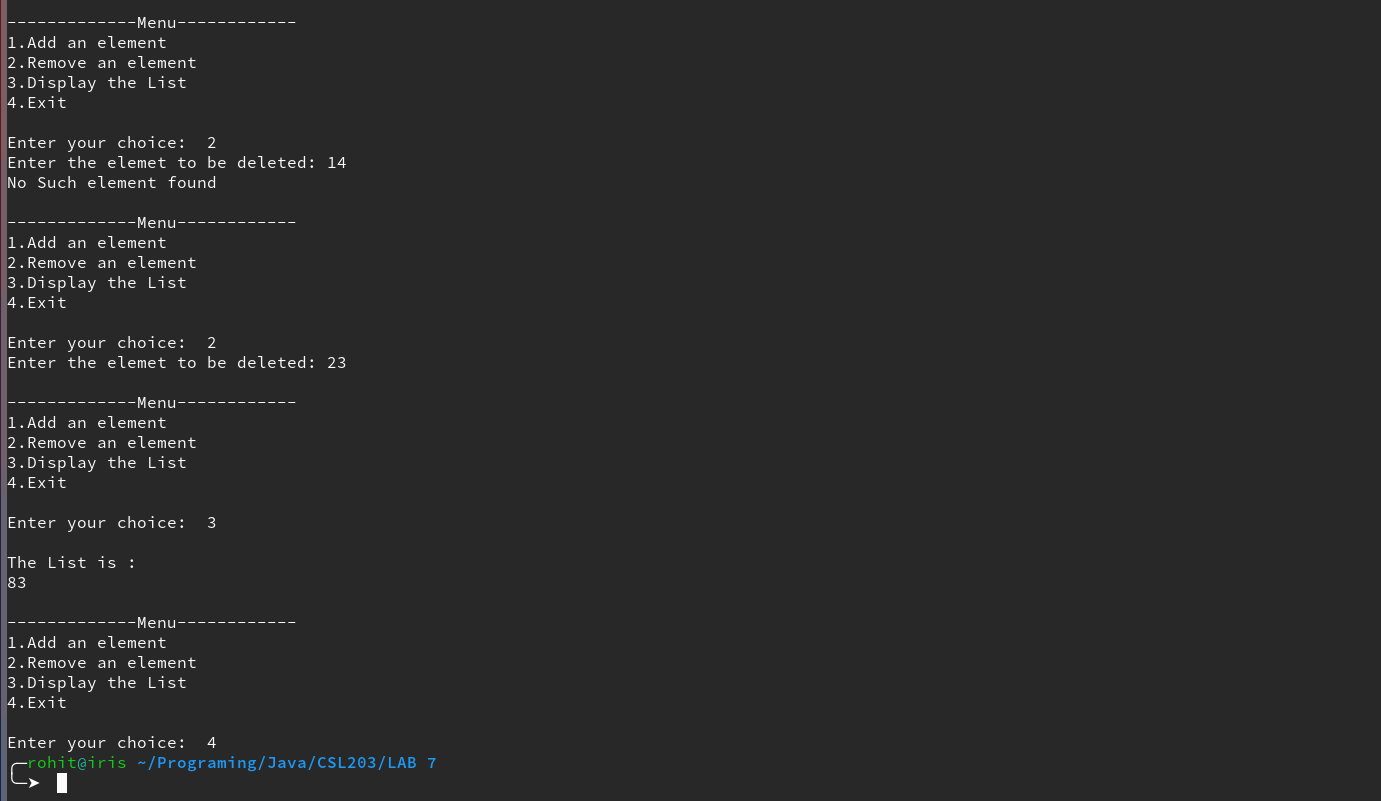
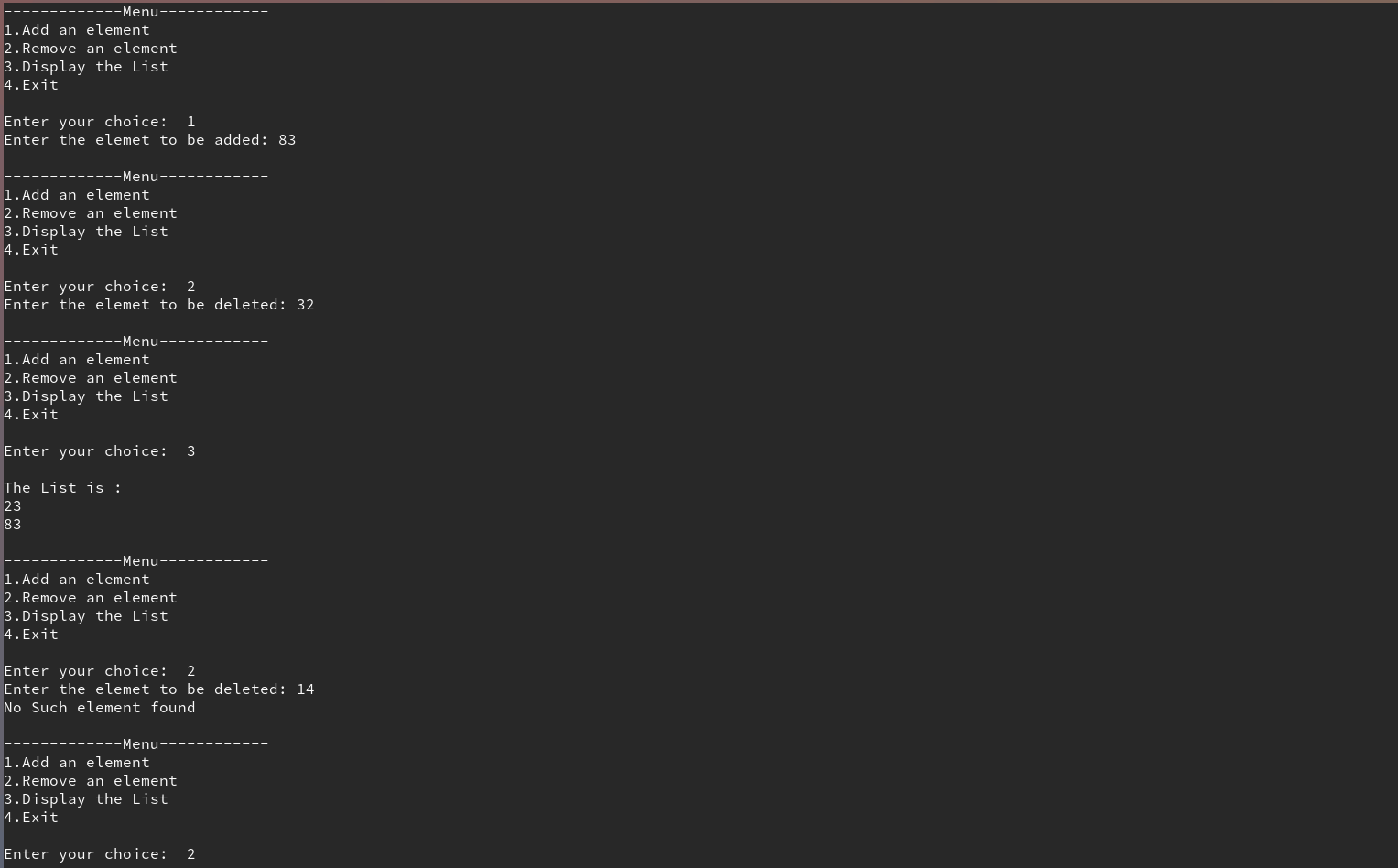
}

}

}

**Sample input/output:**





**Experiment -**

**Aim:** Java Program to implement Binary Search algorithm

**Concepts Used:** ArrayList, BinarySearch

**Algorithm** BinarySearch

Input: the array, the starting index the ending index and the element to be searched

Output: the index of the element, -1 if the element doesn’t exist

Steps

Start

if(start<=last) then

mid = (start+last)/2

if(a[mid] == elem) then

return mid

else if(a[mid]>elem)

return BinarySearch(a,start, mid,elem)

else

return BinarySearch(a,mid+1,last,elem)

endif

else

return -1

endif

Stop

**Program Code:**

/\* Binary search algorithm implementation in java

\* Done By: Rohit Karunakaran

\*\*/

import java.util.ArrayList;

import java.util.StringTokenizer;

import java.io.\*;

class BinarySearch

{

//Recursive binary search funtion

static int binarySearch(ArrayList<Integer> a,int elem,int beg,int last)

{

int mid = (beg+last)/2;

if(beg<=last)

{

if(a.get(mid)==elem)

return mid;

else if(a.get(mid)>elem)

return binarySearch(a,elem,beg,mid);

else

return binarySearch(a,elem,mid+1,last);

}

else

{

return -1; //If the element is not found it will return -1

}

}

public static void main(String args[]) throws IOException

{

ArrayList<Integer> arr = new ArrayList<Integer>();

int elem=0;

try

{

System.out.print("Enter the elements in the array in ascending order in th form \"1 2 32 65 75 \" \nwith out the quotes: ");

BufferedReader br= new BufferedReader(new InputStreamReader(System.in));

String nums = br.readLine();

StringTokenizer st = new StringTokenizer(nums," ");

while(st.hasMoreTokens())

{

arr.add(Integer.parseInt(st.nextToken()));

}

System.out.print("Enter the element to be searched ");

elem = Integer.parseInt(br.readLine());

//arr.sort();

int index = binarySearch(arr,elem,0,arr.size()-1);

if(index==-1)

{

System.out.println("The element is not found\n");

}

else

{

System.out.println("The element is found at index "+index);

}

}

catch(NumberFormatException e)

{

System.out.println("A Number expected ");

e.printStackTrace();

}

}

}

**Sample input/output**

