|  |  |
| --- | --- |
| **Program No.** | 07 |
| **Roll No.** | 1540 |
| **Topic.** | Linked List |
| **Title of Program.** | Singly Linked List |
| **Objective.** |  |

# Source Code:

***SLL.java***

/\* Name: Atharva Naik Roll No: 1540

Unit 4: List

Program: Singly Linked List

\*/

import java.util.Scanner;

//1. Node Class class Node

{

int data; Node next;

public Node(int d)

{

data=d; next=null;

}

}// end of node class

// 2. List Class class List

{

Node head;

Node tail; int size;

// constructor public List()

{

head = null; tail = null; size = 0;

}

//Return size of the list

public int getSize()

{

Node tmp = head; while(tmp!=null)

{

size++; // incrementing size tmp = tmp.next;

}

return size;

}// end of getSize

//Return node at the end of the list public void InsertEnd(int val)

{

Node x = new Node(val); size++;

if(head == null) //empty list

{

}

else

{

}

head = x; tail = x;

tail.next = x; tail = x;

}// end of InsertEnd

// Insert at the Head

public void InsertStart(int val)

{

Node x = new Node(val); size++;

if(head == null) //empty list

{

}

else

{

}

head = x; tail = x;

x.next = head; head = x;

}// end of InsertStart

// display of list

public void Display()

{

Node tmp = head; // Inititalise tmp to the first node in

the SLL

while(tmp!=null)

{

System.out.print(tmp.data + "->"); //Print data at tmp

tmp = tmp.next; //Shift tmp to the next node

}

System.out.println("End of the List \n");

}//end of display

// Search Node

public void Search(int val)

{

Node tmp = head; int flag = 0; while(tmp!=null)

{

if(tmp.data == val) // search value

{

flag = 1; break;

}

tmp = tmp.next;

}//end of while

if(flag==1)

{

}

else

{

}

System.out.println(val + " is Found");

System.out.println(val + " is not Found");

}//end of search

//Deletion of node

public void Del(int val)

{

//1. Search for the node to be deleted. Node tmp = head;

Node prev = null; int flag = 0;

while(tmp!=null)

{

if(tmp.data == val)

{

flag = 1; break;

}

prev = tmp; tmp = tmp.next;

}//end of while

//2. Unsuccessful Search - prompt & return control if(flag==0)

{

System.out.println("Unsucessful Search!"); return;

}

//3. Sucessful search

if(tmp==head && tmp==tail) //3a.Single node deletion

{

head = null; tail = null;

}

else if(tmp == head) // 3b. Head node deletion

{

head = tmp.next;

}

else if(tmp == tail) // 3c. Tail node deletion

{

tail = prev; tail.next = null;

}

else // 3d. any node deletion in middle

{

prev.next = tmp.next;

}

}//end of delete

}//end of list class

//3.Interface class SLL

{

public static void main(String [] args)

{

Scanner scan = new Scanner(System.in);

//Create an object of list List s = new List();

char ch;

do

{

System.out.println("\*\*\*Singly Linked List\*\*\*"); System.out.println(" 1. Insert at the end of the list "); System.out.println(" 2. Size of the list "); System.out.println(" 3. Display the list "); System.out.println(" 4. Search in the list "); System.out.println(" 5. Insert at the Start of the list "); System.out.println(" 6. Deletion of Node");

System.out.println(" Enter your choice: "); int choice = scan.nextInt();

switch(choice)

{

case 1:

case 2:

case 3:

case 4:

case 5:

case 6:

System.out.println("Enter a value at end: "); s.InsertEnd(scan.nextInt());

break;

System.out.println("Size: " + s.getSize()+ "\n"); break;

System.out.println("SLL contains: "); s.Display();

break;

System.out.println("Value to Search: "); s.Search(scan.nextInt());

break;

System.out.println("Enter a value insert at start: "); s.InsertStart(scan.nextInt());

break;

System.out.println("Enter a value for deletion: "); s.Del(scan.nextInt());

break;

default:

System.out.println("Incorrect Choice");

}// end of the switch

System.out.println("Do you want to continue?? (Type: Y or N)"); ch = scan.next().charAt(0);

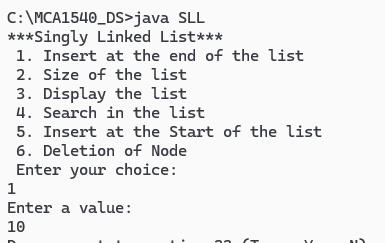
}

while(ch == 'y' || ch == 'Y'); // end of do while

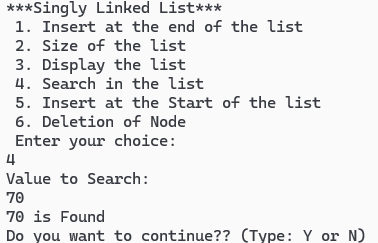
}//end of main

}// end of class SLL

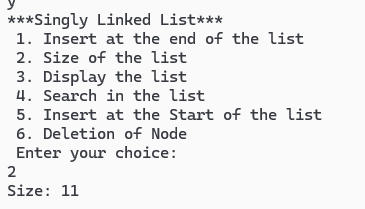
# Output Screenshot

**Insert at end of list**

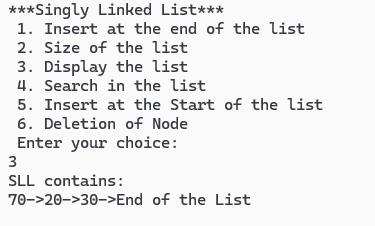
# Insert at start of list

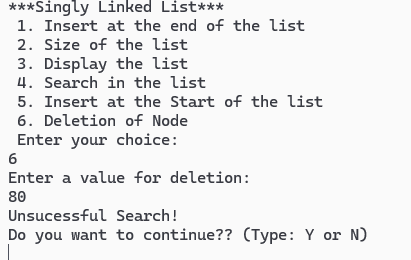
**Search Node Found**

# Not Found

**Count/Size of List**

# Display of List



**Deletion of node Not found**

# Found