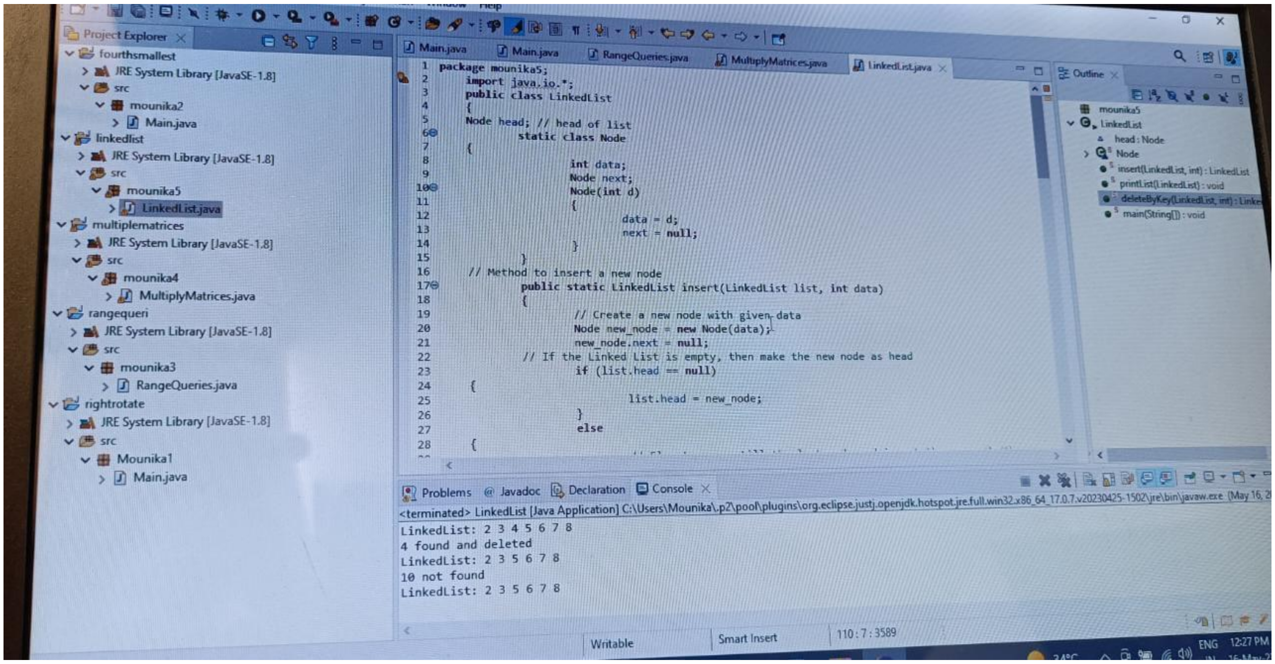
Program 5: first occurrence of a key in a singly linked list

```
package mounika5;
import java.io.*;
public class LinkedList
{
Node head; // head of list
    static class Node
{
            int data;
            Node next;
            Node(int d)
            {
                data = d;
                next = null;
            }
    }
// Method to insert a new node
    public static LinkedList insert(LinkedList list, int data)
    {
            // Create a new node with given data
            Node new_node = new Node(data);
            new_node.next = null;
        // If the Linked List is empty, then make the new node as head
            if (list.head == null)
{
                list.head = new_node;
            }
            else
{
                // Else traverse till the last node and insert the new_node there
                Node last = list.head;
                while (last.next != null)
{
                    last = last.next;
                }
            // Insert the new_node at last node
                last.next = new_node;
            }
            return list;
    }
    public static void printList(LinkedList list)
    {
            Node currNode = list.head;
            System.out.print("LinkedList: ");
            // Traverse through the LinkedList
            while (currNode != null)
{
                // Print the data at current node
                System.out.print(currNode.data + " ");
                // Go to next node
                currNode = currNode.next;
            }
```

```
                System.out.println();
        }
        // Method to delete a node in the LinkedList by KEY
        public static LinkedList deleteByKey(LinkedList list, int key)
        {
                // Store head node
                Node currNode = list.head, prev = null;
                if (currNode != null && currNode.data == key)
    {
                    list.head = currNode.next; // Changed head
                    System.out.println(key + " found and deleted");
                    return list;
                }
                while (currNode != null && currNode.data != key)
    {
                    prev = currNode;
                    currNode = currNode.next;
                }
                if (currNode != null)
    {
                    prev.next = currNode.next;
                    System.out.println(key + " found and deleted");
                }
                if (currNode == null)
    {
                    System.out.println(key + " not found");
                }
                return list;
        }
        // method to create a Singly linked list with n nodes
        public static void main(String[] args)
        {
                /* Start with the empty list. */
                LinkedList list = new LinkedList();
                // Insert the values
                list = insert(list, 1);
                list = insert(list, 2);
                list = insert(list, 3);
                list = insert(list, 4);
                list = insert(list, 5);
                list = insert(list, 6);
                list = insert(list, 7);
                list = insert(list, 8);
                // Print the LinkedList
                printList(list);
                // Delete node with value 1
                deleteByKey(list, 1);
                // Print the LinkedList
                printList(list);
                // Delete node with value 4
                deleteByKey(list, 4);
                // Print the LinkedList
                printList(list);
                // Delete node with value 10
                deleteByKey(list, 10);
```

```
                        // Print the LinkedList
                        printList(list);
        }
    }
```



```
package mounika5;
    import java.io.*;
    public class LinkedList
    {
    Node head; // head of list
            static class Node
    {
                    int data;
                    Node next;
                    Node(int d)
                    {
                            data = d;
                            next = null;
                    }
            }
    // Method to insert a new node
            public static LinkedList insert(LinkedList list, int data)
            {
                    // Create a new node with given data
                    Node new_node = new Node(data);
                    new_node.next = null;
            // If the Linked List is empty, then make the new node as head
                    if (list.head == null)
    {
                            list.head = new_node;
                    }
                    else
    {
```

```
<terminated> LinkedList [Java Application] C:\Users\Mounika\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (May 16, 2
LinkedList: 2 3 4 5 6 7 8
4 found and deleted
LinkedList: 2 3 5 6 7 8
10 not found
LinkedList: 2 3 5 6 7 8
```