**Assisted Practice: 4.5 Singly Linked List**

This section will guide you to:

* Write a program in Java to delete the first occurrence of a key in a singly linked list
* Use Eclipse (the popular text editor for Java programs)
* Push code to Git

This lab has three subsections, namely:

* + 1. Creating a new project in Eclipse
    2. Writing the program in Java to understand the working of the singly linked list
    3. Pushing the code to your GitHub repositories

**Step 4.5.1:** Creating a new project in Eclipse

* Open Eclipse
* Go to File -> New -> Project -> Java Project -> Next
* Type in any project name and click on Finish
* Select your project and go to File -> New -> Class
* Enter **LinkedList** in class name, check the checkbox **public static void main(String[] args)**, and click on Finish

**Step 4.5.2:** Writing the program in Java to understand the working of the singly linked list

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);

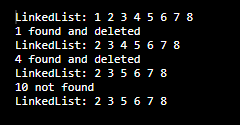
// Print the LinkedList

printList(list);

}

}

**Output:**



**Step 4.5.3:** Pushing the code to your GitHub repositories

* Open your command prompt and navigate to the folder where you have created your files.

**cd <folder path>**

* Initialize your repository using the following command:

**git init**

* Add all the files to your git repository using the following command:

**git add .**

* Commit the changes using the following command:

**git commit . -m “Changes have been committed.”**

* Push the files to the folder you initially created using the following command:

**git push -u origin master**