import java.lang.\*;

class LinkList {

Node head;

static class Node {

int data;

Node next;

Node(int val) { // Constructor

data = val;

next = null;

}

}

public void printList() {

Node n = head;

while(n != null) {

System.out.print(n.data + "-> ");

n = n.next;

}

}

// Add new element at the begininning

public void push(int data) {

Node ad = new Node(data);

ad.next = head;

head = ad; // Insert at the beginning

}

// Add node after a specific node

public void afterNode(Node prev\_Node, int val) {

if(prev\_Node == null) { // if that node is not in list

return ;

}

Node nd = new Node(val);

nd.next = prev\_Node.next;

prev\_Node.next = nd;

}

// At the end of Linked list

public void end\_of\_List(int data) {

if(head == null) {

return; // Empty list

}

else {

Node n = head;

while(n.next != null) {

n = n.next;

}

Node nd = new Node(data);

n.next = nd;

}

}

// Delete at given position

public void deletePosition(int pos) {

Node n;

n = head;

if(pos == 1) { // if 1st node to be deleted

n = n.next;

return;

}

else {

int i;

for(i = 0; i < pos -1 && n != null; i++) {

n = n.next;

}

if(n == null || n.next == null) {

return;

}

else {

Node ne = n.next.next;

n.next = ne;

}

}

}

// DeleteNode a given key

public void DeleteNode(int key) {

Node n, prev;

n = head;

prev = null;

if(n != null && n.data == key) { // key holding is head

head = n.next;

return;

}

else {

while(n != null && n.data != key) {

prev = n;

n = n.next;

}

if(n == null) { // key not found

return ;

}

else {

prev.next = n.next;

}

}

}

public int iter\_length() {

Node nd = head;

int len = 0;

if(nd == null) { // If list is empty

return 0;

}

else {

while(nd != null) {

len++;

nd = nd.next;

}

}

return len;

}

public int recur\_length(Node n) {

if(n == null) {

return 0;

}

else {

return recur\_length(n.next) + 1;

}

}

public static void main(String[] args) {

LinkList llist = new LinkList();

llist.head = new Node(1);

Node second = new Node(2);

Node third = new Node(3);

llist.head.next = second;

second.next = third;

System.out.println(" New List");

llist.printList();

// Add new element at the begininning

llist.push(5);

System.out.println("\n New node after push");

llist.printList();

System.out.println(" \n After adding at specific node");

llist.afterNode(third, 10);

llist.printList();

// Add node at the end

llist.end\_of\_List(15);

System.out.println("\n After Adding at the end");

llist.printList();

// After Deletion

llist.DeleteNode(10);

System.out.println("\n After Deletion of key 10");

llist.printList();

// Deletion from position

llist.deletePosition(3);

System.out.println("\n After Deletion of pos 3");

llist.printList();

// Find iterative length of list

System.out.println("\n Length of List \n" +llist.iter\_length() + "");

// Find Recursive length of List

System.out.println("\n Length of List from Recursion \n " + llist.recur\_length(llist.head) + " ");

}

}