Assignment no 3

cs231160

Suresh kumar

3d

*/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
  
 Online Java Compiler.  
 Code, Compile, Run and Debug java program online.  
 Write your code in this editor and press "Run" button to execute it.  
  
 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/*public class Sll  
{  
 public static Node *head* = null;  
 static class Node{  
 int data;  
 Node next;  
  
 Node(int data){  
 this.data = data;  
 this.next = null;  
 }  
  
 }  
 public static void add(int element){  
 Node newNode = new Node(element);  
  
 if (*head* == null){  
 *head* = newNode;  
 return;  
 }  
 newNode.next = *head*;  
 *head* = newNode;  
  
 }  
 public static void printData(){  
 if (*head* == null){  
 System.*out*.print("This is Empty");  
 return;  
 }  
 int index = 0;  
 Node temp = *head*;  
 while(temp!=null){  
 System.*out*.print(temp.data+" --> ");  
 temp = temp.next;  
 index++;  
 }  
 Node current = *head*;  
 int middleValue = index/2;  
 for (int i = 0; i<middleValue; i++){  
 current = current.next;  
 }  
 System.*out*.println("The middle value is "+current.data);  
 System.*out*.print("The length is "+index);  
 System.*out*.println();  
 }  
 public void reverse() {  
 Node prev = null;  
 Node current = *head*;  
  
 while (current != null) {  
 Node next = current.next;  
 current.next = prev;  
 prev = current;  
 current = next;  
 }  
  
  
 while(prev!=null){  
 System.*out*.print(prev.data+" --> ");  
 prev = prev.next;  
  
 }  
 // 'prev' will be the new head after reversal  
 }  
  
 public static void remnoveDuplicate(){  
 Node current = *head*;  
  
  
 while (current != null && current.next != null) {  
 if (current.data == current.next.data) {  
 current.next = current.next.next;  
 } else {  
 current = current.next;  
 }  
 }  
  
 }  
// public void printList() {  
// Node temp = head;  
// while (temp != null) {  
// System.out.print(temp.data + " ");  
// temp = temp.next;  
// }  
// System.out.println();  
// }  
 public static Node MErgeListNode(Node n1, Node n2){  
 Node dummy = new Node(0);  
 Node curr = dummy;  
 while (n1!=null && n2!=null){  
 if (n1.data <= n2.data){  
 curr.next = n1;  
 n1 = n1.next;  
 }  
 else {  
 curr.next = n2;  
 n2 = n2.next;  
 }  
 curr = curr.next;  
  
 }  
 if (n1 != null) {  
 curr.next = n1;  
 } else {  
 curr.next = n2;  
 }  
 return dummy.next;  
 }  
 public static void printList(Node head) {  
 Node temp = head;  
 if (head==null){  
 System.*out*.println("Linked List is empty");  
 return;  
 }  
 while (temp != null) {  
 System.*out*.print(temp.data + " ");  
 temp = temp.next;  
 }  
 System.*out*.println();  
 }  
 public static Node deleteList(Node head){  
 head = null;  
 return head;  
 }  
 public static void main(String[] args) {  
// Sll sll = new Sll();  
// sll.add(20);  
// sll.add(30);  
// sll.add(40);  
// sll.add(40);  
// sll.add(50);  
// sll.add(60);  
// sll.add(60);  
// sll.printData();  
// sll.reverse();  
// sll.remnoveDuplicate();  
// sll.printList();  
 Node n1 = new Node(1);  
 n1.next = new Node(2);  
 n1.next.next = new Node(3);  
  
 Node n2 = new Node(1);  
 n2.next = new Node(3);  
 n2.next.next = new Node(4);  
  
// Node mergeList = MErgeListNode(n1, n2);  
// printList(mergeList);  
  
 *printList*(n2);  
 System.*out*.println(*deleteList*(n2));  
// printList(n2);  
 }  
}







