1. Implement singly linked list in java

public class LinkedList {

Node head;

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)

{

Node new\_node = new Node(data);

if (list.head == null) {

list.head = new\_node;

}

else {

Node last = list.head;

while (last.next != null) {

last = last.next;

}

last.next = new\_node;

}

return list;

}

public static void printList(LinkedList list)

{

Node currNode = list.head;

System.out.print("LinkedList: ");

while (currNode != null) {

System.out.print(currNode.data + " ");

currNode = currNode.next;

}

}

public static void main(String[] args)

{

LinkedList list = new LinkedList();

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);

printList(list);

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java LinkedList

LinkedList: 1 2 3 4 5 6 7 8

2.Implement Doubly linked list in java

C:\Users\mujah\Downloads\Algorithm and Data Structure>java LinkedList

class node {

node prev;

int data;

node next;

// A constructor is called here

node(int value)

{

// By default previous pointer is

// pointed to NULL

prev = null;

// Value is assigned to the data

data = value;

// By default next pointer is pointed

// to NULL

next = null;

}

}

class LinkedList {

// Declaring an empty doubly linked list

static node head = null;

static node tail = null;

static void insertAtBeginning(int data)

{

node temp = new node(data);

if (head == null) {

head = temp;

tail = temp;

}

else {

temp.next = head;

head.prev = temp;

head = temp;

}

}

static void insertAtEnd(int data)

{

node temp = new node(data);

if (tail == null) {

head = temp;

tail = temp;

}

else {

tail.next = temp;

temp.prev = tail;

tail = temp;

}

}

static void insertAtPosition(int data, int position)

{

node temp = new node(data);

if (position == 1) {

insertAtBeginning(data);

}

else {

node current = head;

int currPosition = 1;

while (current != null

&& currPosition < position) {

current = current.next;

currPosition++;

}

if (current == null) {

insertAtEnd(data);

}

else {

temp.next = current;

temp.prev = current.prev;

current.prev.next = temp;

current.prev = temp;

}

}

}

static void deleteAtBeginning()

{

if (head == null) {

return;

}

if (head == tail) {

head = null;

tail = null;

return;

}

node temp = head;

head = head.next;

head.prev = null;

temp.next = null;

}

static void deleteAtEnd()

{

if (tail == null) {

return;

}

if (head == tail) {

head = null;

tail = null;

return;

}

node temp = tail;

tail = tail.prev;

tail.next = null;

temp.prev = null;

}

static void deleteAtSpecificPosition(int pos)

{

if (head == null) {

return;

}

if (pos == 1) {

deleteAtBeginning();

return;

}

node current = head;

int count = 1;

while (current != null && count != pos) {

current = current.next;

count++;

}

if (current == null) {

System.out.println("Position wrong");

return;

}

if (current == tail) {

deleteAtEnd();

return;

}

current.prev.next = current.next;

current.next.prev = current.prev;

current.prev = null;

current.next = null;

}

static void display(node head)

{

node temp = head;

while (temp != null) {

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

temp = temp.next;

}

System.out.println("NULL");

}

public static void main(String[] args)

{

insertAtEnd(1);

insertAtEnd(2);

insertAtEnd(3);

insertAtEnd(4);

insertAtEnd(5);

System.out.print("After insertion at tail: ");

display(head);

System.out.print("After insertion at head: ");

insertAtBeginning(0);

display(head);

insertAtPosition(6, 2);

System.out.print(

"After insertion at 2nd position: ");

display(head);

deleteAtBeginning();

System.out.print(

"After deletion at the beginning: ");

display(head);

deleteAtEnd();

System.out.print("After deletion at the end: ");

display(head);

deleteAtSpecificPosition(2);

System.out.print(

"After deletion at 2nd position: ");

display(head);

}

}

After insertion at tail: 1 --> 2 --> 3 --> 4 --> 5 --> NULL

After insertion at head: 0 --> 1 --> 2 --> 3 --> 4 --> 5 --> NULL

After insertion at 2nd position: 0 --> 6 --> 1 --> 2 --> 3 --> 4 --> 5 --> NULL

After deletion at the beginning: 6 --> 1 --> 2 --> 3 --> 4 --> 5 --> NULL

After deletion at the end: 6 --> 1 --> 2 --> 3 --> 4 --> NULL

After deletion at 2nd position: 6 --> 2 --> 3 --> 4 --> NULL

3.How to reverse a linked list in java

C:\Users\mujah\Downloads\Algorithm and Data Structure>java LinkedList

class LinkedList

{

static Node head;

static class Node {

int data;

Node next;

Node(int d)

{

data = d;

next = null;

}

}

Node reverse(Node node)

{

Node prev = null;

Node current = node;

Node next = null;

while (current != null) {

next = current.next;

current.next = prev;

prev = current;

current = next;

}

node = prev;

return node;

}

void printList(Node node)

{

while (node != null) {

System.out.print(node.data + " ");

node = node.next;

}

}

public static void main(String[] args)

{

LinkedList list = new LinkedList();

list.head = new Node(85);

list.head.next = new Node(15);

list.head.next.next = new Node(4);

list.head.next.next.next = new Node(20);

System.out.println("Given linked list");

list.printList(head);

head = list.reverse(head);

System.out.println("");

System.out.println("Reversed linked list ");

list.printList(head);

}

}

Given linked list

85 15 4 20

Reversed linked list

20 4 15 85

4. How to merge two linked list in sorted order in java

public class aa {

static class Node{

int data;

Node next;

Node(int data){

this.data=data;

}

}

static Node head1=null,tail1=null,head2=null,tail2=null;

public static void main(String[] args) {

addNodeFirst(10);

addNodeFirst(20);

addNodeFirst(30);

addNodeFirst(40);

addNodeFirst(50);

addNodeFirst(60);

addNodeSecond(5);

addNodeSecond(15);

addNodeSecond(25);

addNodeSecond(35);

addNodeSecond(45);

addNodeSecond(55);

addNodeSecond(65);

System.out.println("List 1 is : ");

printList(head1);

System.out.println("List 2 is : ");

printList(head2);

Node mergeList=mergeList(head1,head2);

System.out.println("Merge List is : ");

printList(mergeList);

}

static Node mergeList(Node h1,Node h2) {

if(h1==null) {

return h2;

}

if(h2==null) {

return h1;

}

Node mergeHead=null,mergeTail=null;

while(h1!=null && h2!=null) {

Node n=null;

if(h1.data<h2.data) {

n=new Node(h1.data);

h1=h1.next;

}else {

n=new Node(h2.data);

h2=h2.next;

}

if(mergeHead==null) {

mergeHead=n;

mergeTail=n;

}else {

mergeTail.next=n;

mergeTail=n;

}

}

if(h1!=null) {

mergeTail.next=h1;

}

if(h2!=null) {

mergeTail.next=h2;

}

return mergeHead;

}

static void printList(Node head) {

Node temp=head;

while(temp!=null) {

System.out.print(temp.data+" ");

temp=temp.next;

}

System.out.println();

}

static void addNodeFirst(int data) {

Node n=new Node(data);

if(head1==null) {

head1=n;

tail1=n;

}else {

tail1.next=n;

tail1=n;

}

}

static void addNodeSecond(int data) {

Node n=new Node(data);

if(head2==null) {

head2=n;

tail2=n;

}else {

tail2.next=n;

tail2=n;

}

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java aa

List 1 is :

10 20 30 40 50 60

List 2 is :

5 15 25 35 45 55 65

Merge List is :

5 10 15 20 25 30 35 40 45 50 55 60 65

5.How to find middle element of linked list in java

public class aa {

static class Node{

int data;

Node next;

Node(int data){

this.data=data;

}

}

static Node head=null,tail=null;

public static void main(String[] agrs) {

addNode(10);

addNode(20);

addNode(30);

addNode(40);

addNode(50);

printList(head);

System.out.println("Middle of Linked List is : ");

findMiddle(head);

}

static void findMiddle(Node head) {

Node slow=head;

Node fast=head;

while(fast!=null && fast.next!=null) {

fast=fast.next.next;

slow=slow.next;

}

System.out.println(slow.data);

}

static void printList(Node head) {

Node temp=head;

while(temp!=null) {

System.out.print(temp.data+" ");

temp=temp.next;

}

System.out.println();

}

static void addNode(int data) {

Node n=new Node(data);

if(head==null) {

head=n;

tail=n;

}else {

tail.next=n;

tail=n;

}

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java aa

10 20 30 40 50

Middle of Linked List is :

30

6. How to detect a loop in linked list in java

public class aa {

static class Node{

int data;

Node next;

Node(int data){

this.data=data;

}

}

static Node head=null;

public static void main(String[] args) {

Node n1=new Node(10);

Node n2=new Node(20);

Node n3=new Node(30);

Node n4=new Node(40);

Node n5=new Node(50);

Node n6=new Node(60);

Node n7=new Node(70);

head=n1;

n1.next=n2;

n2.next=n3;

n3.next=n4;

n4.next=n5;

n5.next=n6;

n6.next=n7;

n7.next=n3;

System.out.println(isLoopExist(head));

}

static boolean isLoopExist(Node head) {

Node slow=head.next;

Node fast=head.next;

while(fast!=null && fast.next!=null) {

if(slow==fast) {

return true;

}

slow=slow.next;

fast=fast.next.next;

}

return false;

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java aa

True

8.How to find nth element from end of linked list

public class aa {

static class Node{

int data;

Node next;

Node(int data){

this.data=data;

}

}

static Node head=null;

public static void main(String[] args) {

Node n1=new Node(10);

Node n2=new Node(20);

Node n3=new Node(30);

Node n4=new Node(40);

Node n5=new Node(50);

Node n6=new Node(60);

Node n7=new Node(70);

head=n1;

n1.next=n2;

n2.next=n3;

n3.next=n4;

n4.next=n5;

n5.next=n6;

n6.next=n7;

n7.next=n6;

findFirstNodeInLoop(head);

}

static void findFirstNodeInLoop(Node head) {

Node slow=head.next;

Node fast=head.next.next;

boolean isLoopExist=false;

while(fast!=null && fast.next!=null) {

if(slow==fast) {

isLoopExist=true;

break;

}

slow=slow.next;

fast=fast.next.next;

}

if(isLoopExist) {

slow=head;

while(slow!=fast) {

slow=slow.next;

fast=fast.next;

}

System.out.println("Firs node of the loop is "+slow.data);

}

else {

System.out.println("Loop is not exist in the LinkedList.");

}

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java aa

Firs node of the loop is 60

9.How to check if linked list is palindrome in java

import java.util.\*;

class aa{

static class Node{

int data;

Node next;

Node(int data){

this.data=data;

}

}

static Node head=null,tail=null;

static private Scanner sc=new Scanner(System.in);

public static void main(String[] agrs) {

addNode(10);

addNode(20);

addNode(30);

addNode(40);

addNode(50);

addNode(60);

addNode(70);

printList(head);

System.out.println("Enter position of elemet for last");

int k=sc.nextInt();

findKthElement(head,k);

}

static void findKthElement(Node head,int k) {

if(head==null || k<0) {

System.out.println("Sorry Bro....");

return;

}

Node slow=head;

Node fast=head;

for(int i=1;i<k;i++) {

fast=fast.next;

}

while(fast!=null && fast.next!=null) {

slow=slow.next;

fast=fast.next;

}

System.out.println("Kth Element for last is "+slow.data);

}

static void printList(Node head) {

Node temp=head;

while(temp!=null) {

System.out.print(temp.data+" ");

temp=temp.next;

}

System.out.println();

}

static void addNode(int data) {

Node n=new Node(data);

if(head==null) {

head=n;

tail=n;

}else {

tail.next=n;

tail=n;

}

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java aa

10 20 30 40 50 60 70

Enter position of elemet for last

6

Kth Element for last is 20

10.Add two numbers represented by linked list in java

import java.util.\*;

class aa{

static class Node{

int data;

Node next;

Node(int data){

this.data=data;

}

}

static Node head1=null,tail1=null,head2=null,tail2=null;

public static void main(String[] args) {

addNodeFirst(1);

addNodeFirst(2);

addNodeFirst(3);

addNodeFirst(4);

addNodeFirst(5);

addNodeSecond(1);

addNodeSecond(2);

addNodeSecond(3);

addNodeSecond(4);

addNodeSecond(5);

addNodeSecond(6);

System.out.println("List 1 is : ");

printList(head1);

System.out.println("List 2 is : ");

printList(head2);

System.out.println();

System.out.println("Addition of this two list are");

Node head=addList(head1,head2);

printList(head);

}

static Node addList(Node h1,Node h2) {

if(h1==null) {

return h2;

}

if(h2==null) {

return h1;

}

h1=reverseList(h1);

h2=reverseList(h2);

Node head=null;

int carry=0;

while(h1!=null && h2!=null) {

int val1=h1.data;

int val2=h2.data;

int ans=val1+val2+carry;

int val=ans%10;

carry=ans/10;

Node n=new Node(val);

if(head==n) {

head=n;

}else {

n.next=head;

head=n;

}

h1=h1.next;

h2=h2.next;

}

while(h1!=null) {

int val=h1.data+carry;

int ans=val%10;

carry=val/10;

Node n=new Node(ans);

n.next=head;

head=n;

h1=h1.next;

}

while(h2!=null) {

int val=h2.data+carry;

int ans=val%10;

carry=val/10;

Node n=new Node(ans);

n.next=head;

head=n;

h2=h2.next;

}

return head;

}

static Node reverseList(Node head) {

Node temp=head;

Node pre=null;

while(temp!=null) {

Node t=temp.next;

temp.next=pre;

pre=temp;

temp=t;

}

return pre;

}

static void printList(Node head) {

Node temp=head;

while(temp!=null) {

System.out.print(temp.data+" ");

temp=temp.next;

}

System.out.println();

}

static void addNodeFirst(int data) {

Node n=new Node(data);

if(head1==null) {

head1=n;

tail1=n;

}else {

tail1.next=n;

tail1=n;

}

}

static void addNodeSecond(int data) {

Node n=new Node(data);

if(head2==null) {

head2=n;

tail2=n;

}else {

tail2.next=n;

tail2=n;

}

}

}

C:\Users\mujah\Downloads\Algorithm and Data Structure>java aa

List 1 is :

1 2 3 4 5

List 2 is :

1 2 3 4 5 6

Addition of this two list are

1 3 5 8 0 1