2. Write a Java program to find the maximum and minimum value node from a circular linked list

Algorithm  
Define a Node class which represents a node in the list. It has two properties data and next which will point to the next node.  
Define another class for creating the circular linked list and it has two nodes: head and tail.  
minNode() will print out minimum value node:  
Define variable min and initialize with head's data.  
Current will point to head.  
Iterate through the list by comparing each node's data with min.  
If min > current's data then min will hold current's data.  
At the end of the list, variable min will hold the minimum value node.  
Print the min value.

maxNode() will prints out maximum value node:  
Define variable max and initialize with head's data.  
Current will point to head.  
Iterate through the list by comparing each node's data with max.  
If max > current's data then max will hold current's data.  
At the end of the list, variable max will hold the maximum value node.  
Print the max value.

**Code:**

public class Main{

public class Node{

int data;

Node next;

public Node(int data) {

this.data = data;

}

}

public Node head = null;

public Node tail = null;

public void add(int data){

Node newNode = new Node(data);

if(head == null) {

head = newNode;

tail = newNode;

newNode.next = head;

}

else {

tail.next = newNode;

tail = newNode;

tail.next = head;

}

}

public void minNode() {

Node current = head;

int min = head.data;

if(head == null) {

System.out.println("List is empty");

}

else {

do{

if(min > current.data) {

min = current.data;

}

current= current.next;

}while(current != head);

System.out.println("Minimum value node in the list: "+ min);

}

}

public void maxNode() {

Node current = head;

int max = head.data;

if(head == null) {

System.out.println("List is empty");

}

else {

do{

if(max < current.data) {

max = current.data;

}

current= current.next;

}while(current != head);

System.out.println("Maximum value node in the list: "+ max);

}

}

public static void main(String[] args) {

Main cl = new Main();

cl.add(5);

cl.add(25);

cl.add(15);

cl.add(2);

cl.minNode();

cl.maxNode();

}

}

**Output:**

