public class Doubly {

Node head= null;

Node tail= null;

class Node{

int value=0;

Node next= null;

Node prev= null;

Node(int value){

this.value= value;

}

}

public void insert(int value) {

Node n= new Node(value);

if(head==null) {

head= n;

tail= n;

n.next= null;

n.prev= null;

}

else {

tail.next= n;

n.prev= tail;

tail=n;

tail.next= null;

}

}

public void insertAtPosition(int value, int pos) {

Node n= new Node(value);

if(head==null) {

head= n;

tail= n;

n.next= null;

n.prev= null;

}

else {

Node temp=null;

Node cur= head;

for(int i=0; i<pos; i++) {

temp= cur;

cur= cur.next;

}

temp.next= n;

n.prev= temp;

n.next= cur;

cur.prev= n;

}

}

public void deletetAtPosition(int pos) {

if(head==null) {

System.out.print("list is empty");

}

else {

Node temp=null;

Node cur= head;

for(int i=0; i<pos; i++) {

temp= cur;

cur= cur.next;

}

temp.next= cur.next;

cur.next.prev= temp;

}

}

public void display() {

Node current = head;

if(current==null) {

System.out.print("empty list");

}

else {

while(current!=null) {

System.out.println(current.value);

current= current.next;

}

}

}

public void revdisplay() {

Node current = tail;

if(current==null) {

System.out.print("empty list");

}

else {

while(current!=null) {

System.out.println(current.value);

current= current.prev;

}

}

}

public static void main(String[] args) {

Doubly d= new Doubly();

d.insert(2);

d.insert(3);

d.insert(5);

d.insert(23);

d.display();

d.revdisplay();

}

}

Method 2

**import** java.util.Scanner;

**class** Node {

**int** data;

Node prev;

Node next;

Node(**int** data) {

**this**.data = data;

**this**.prev = **null**;

**this**.next = **null**;

}

}

**class** Doubly {

Node head;

**void** insertAtFront(**int** data) {

Node newNode = **new** Node(data);

**if** (head == **null**) {

head = newNode;

} **else** {

newNode.next = head;

head.prev = newNode;

head = newNode;

}

System.***out***.println("Inserted " + data + " at the front.");

}

**void** insertAtEnd(**int** data) {

Node newNode = **new** Node(data);

**if** (head == **null**) {

head = newNode;

} **else** {

Node temp = head;

**while** (temp.next != **null**) {

temp = temp.next;

}

temp.next = newNode;

newNode.prev = temp;

}

System.***out***.println("Inserted " + data + " at the end.");

}

**void** insertAfter(**int** data, **int** afterData) {

Node newNode = **new** Node(data);

**if** (head == **null**) {

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

**return**;

}

Node current = head;

**while** (current != **null** && current.data != afterData) {

current = current.next;

}

**if** (current == **null**) {

System.***out***.println(afterData + " not found in the list.");

} **else** {

Node nextNode = current.next;

current.next = newNode;

newNode.prev = current;

newNode.next = nextNode;

**if** (nextNode != **null**) {

nextNode.prev = newNode;

}

System.***out***.println("Inserted " + data + " after " + afterData + ".");

}

}

**void** deleteAtFront() {

**if** (head == **null**) {

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

**return**;

}

head = head.next;

**if** (head != **null**) {

head.prev = **null**;

}

System.***out***.println("Deleted the node at the beginning.");

}

**void** deleteAtEnd() {

**if** (head == **null**) {

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

**return**;

}

**if** (head.next == **null**) {

head = **null**;

System.***out***.println("Deleted the only node in the list.");

**return**;

}

Node current = head;

**while** (current.next != **null**) {

current = current.next;

}

Node prevNode = current.prev;

prevNode.next = **null**;

System.***out***.println("Deleted the node at the end.");

}

**void** deleteByData(**int** data) {

**if** (head == **null**) {

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

**return**;

}

**if** (head.data == data) {

head = head.next;

**if** (head != **null**) {

head.prev = **null**;

}

System.***out***.println("Deleted the node with data " + data + ".");

**return**;

}

Node current = head;

**while** (current != **null** && current.data != data) {

current = current.next;

}

**if** (current == **null**) {

System.***out***.println("Node with data " + data + " not found in the list.");

} **else** {

Node prevNode = current.prev;

Node nextNode = current.next;

prevNode.next = nextNode;

**if** (nextNode != **null**) {

nextNode.prev = prevNode;

}

System.***out***.println("Deleted the node with data " + data + ".");

}

}

Integer search(**int** data) {

**if** (head == **null**) {

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

**return** **null**;

}

Node current = head;

**int** position = 1;

**while** (current != **null**) {

**if** (current.data == data) {

System.***out***.println(data + " found at position " + position + ".");

**return** position;

}

current = current.next;

position++;

}

System.***out***.println(data + " not found in the list.");

**return** **null**;

}

**void** display() {

**if** (head == **null**) {

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

**return**;

}

Node current = head;

System.***out***.print("Doubly Linked List: ");

**while** (current != **null**) {

System.***out***.print(current.data + " <-> ");

current = current.next;

}

System.***out***.println("null");

}

**public** **static** **void** main(String[] args) {

Scanner scanner = **new** Scanner(System.***in***);

Doubly dll = **new** Doubly();

**while** (**true**) {

System.***out***.println("\nMenu:");

System.***out***.println("1. Insert at the front");

System.***out***.println("2. Insert at the end");

System.***out***.println("3. Insert after a specified node");

System.***out***.println("4. Delete at the beginning");

System.***out***.println("5. Delete at the end");

System.***out***.println("6. Delete by data");

System.***out***.println("7. Search");

System.***out***.println("8. Display the list");

System.***out***.println("9. Exit");

System.***out***.print("Enter your choice: ");

**int** choice = scanner.nextInt();

**switch** (choice) {

**case** 1:

System.***out***.print("Enter data to insert: ");

**int** data1 = scanner.nextInt();

dll.insertAtFront(data1);

**break**;

**case** 2:

System.***out***.print("Enter data to insert: ");

**int** data2 = scanner.nextInt();

dll.insertAtEnd(data2);

**break**;

**case** 3:

System.***out***.print("Enter data to insert: ");

**int** data3 = scanner.nextInt();

System.***out***.print("Enter data to insert after: ");

**int** afterData = scanner.nextInt();

dll.insertAfter(data3, afterData);

**break**;

**case** 4:

dll.deleteAtFront();

**break**;

**case** 5:

dll.deleteAtEnd();

**break**;

**case** 6:

System.***out***.print("Enter data to delete: ");

**int** data6 = scanner.nextInt();

dll.deleteByData(data6);

**break**;

**case** 7:

System.***out***.print("Enter data to search: ");

**int** data7 = scanner.nextInt();

dll.search(data7);

**break**;

**case** 8:

dll.display();

**break**;

**case** 9:

scanner.close();

System.*exit*(0);

**default**:

System.***out***.println("Invalid choice. Please try again.");

}

}

}

}