PRACTICE-PROJECT7

TO TRAVERSE A DOUBLY LINKED LIST IN THE FORWARD AND BACKWARD DIRECTONS –

**package** practice;

**public** **class** DoublyLinkedList {

**static** **class** Node {

**int** data;

Node prev;

Node next;

Node(**int** data) {

**this**.data = data;

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

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

}

}

Node head = **null**;

Node tail = **null**;

**public** **void** insert(**int** data) {

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

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

head = newNode;

tail = newNode;

}

**else** {

newNode.prev = tail;

tail.next = newNode;

tail = newNode;

}

}

**public** **void** traverseForward() {

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

System.***out***.println("The list is empty.");

}

**else** {

System.***out***.println("Traversing the list in forward direction: ");

Node currentNode = head;

**do** {

System.***out***.print(currentNode.data+" ");

currentNode = currentNode.next;

} **while** (currentNode != **null**);

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

}

}

**public** **void** traverseBackward() {

**if**(tail == **null**) {

System.***out***.println("The list is empty.");

}

**else** {

System.***out***.println("Traversing the list in backward direction: ");

Node currentNode = tail;

**do** {

System.***out***.print(currentNode.data+" ");

currentNode = currentNode.prev;

} **while** (currentNode != **null**);

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

}

}

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

DoublyLinkedList dl = **new** DoublyLinkedList();

dl.insert(12);

dl.insert(7);

dl.insert(10);

dl.insert(3);

dl.insert(45);

dl.traverseForward();

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

dl.traverseBackward();

}

}