**Phase 1 Practice Project – Assisted Practice**

**7 . Writing a program in java to traverse a doubly linked list in the forward and backward directions**

**Source Code:**

**package** slm3;

**class** Node {

**int** data;

Node next;

Node prev;

Node(**int** data) {

**this**.data = data;

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

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

}

}

**class** DoublyLinkedList {

Node head;

DoublyLinkedList() {

head = **null**;

}

// Function to insert a new node at the end of the doubly linked list

**void** append(**int** data) {

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

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

head = newNode;

} **else** {

Node current = head;

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

current = current.next;

}

current.next = newNode;

newNode.prev = current;

}

}

// Function to traverse the doubly linked list in the forward direction

**void** forwardTraversal() {

Node current = head;

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

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

current = current.next;

}

}

// Function to traverse the doubly linked list in the backward direction

**void** backwardTraversal() {

Node current = head;

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

current = current.next;

}

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

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

current = current.prev;

}

}

}

**public** **class** DoublyLinked {

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

DoublyLinkedList list = **new** DoublyLinkedList();

list.append(30);

list.append(32);

list.append(33);

list.append(74);

list.append(85);

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

list.forwardTraversal();

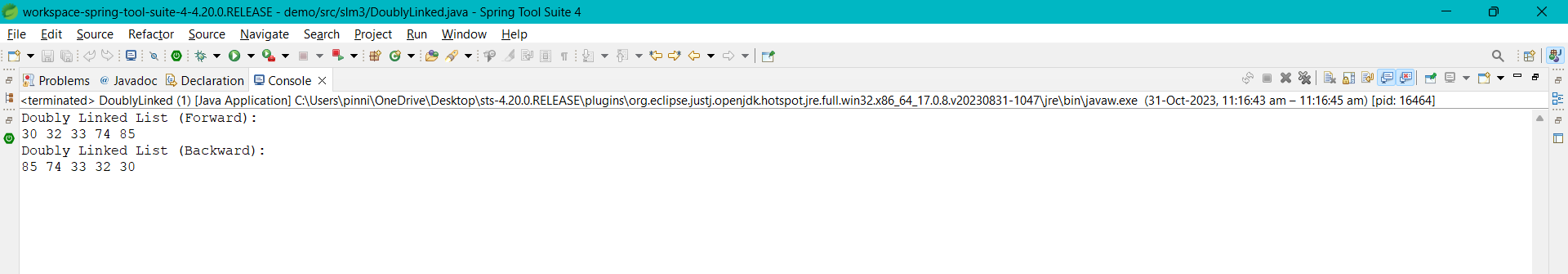
System.***out***.println("\nDoubly Linked List (Backward):");

list.backwardTraversal();

}

}

**Output:**

****