**Write a simple code to identify given linked list is palindrome or not by using stack. First take a Stack. Traverse through each node of the linked list and push each node value to Stack. Once the traversal & copying is done, iterate through linked list from head node again. In each iteration, pop one stack element and compare with node value in respective iteration. It is expected to match stack popped value with node value. In case of all matches, its a palindrome. Any one element mismatch makes it not a palindrome.**

**package** Tuesday;

**import**java.util.Stack;

**publicclass**LinkedListPalindromeUsingStack {

**publicstaticvoid** main(String[] a){

Node n1 = **new** Node(10);

Node n2 = **new** Node(28);

Node n3 = **new** Node(15);

Node n4 = **new** Node(25);

Node n5 = **new** Node(10);

n1.next = n2;

n2.next = n3;

n3.next = n4;

n4.next = n5;

**boolean**result = *isPalindrome*(n1);

System.***out***.println("Is it palindrome: "+result);

}

**staticclass** Node {

**int**data;

Node next;

Node(**int**tmp) {

data = tmp;

}

}

**staticboolean**isPalindrome(Node head) {

Node tempNode = head;

Stack<Integer>stack = **new** Stack<Integer>();

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

stack.push(tempNode.data);

tempNode = tempNode.next;

}

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

**if**(head.data != stack.pop()) {

**return**Boolean.***FALSE***;

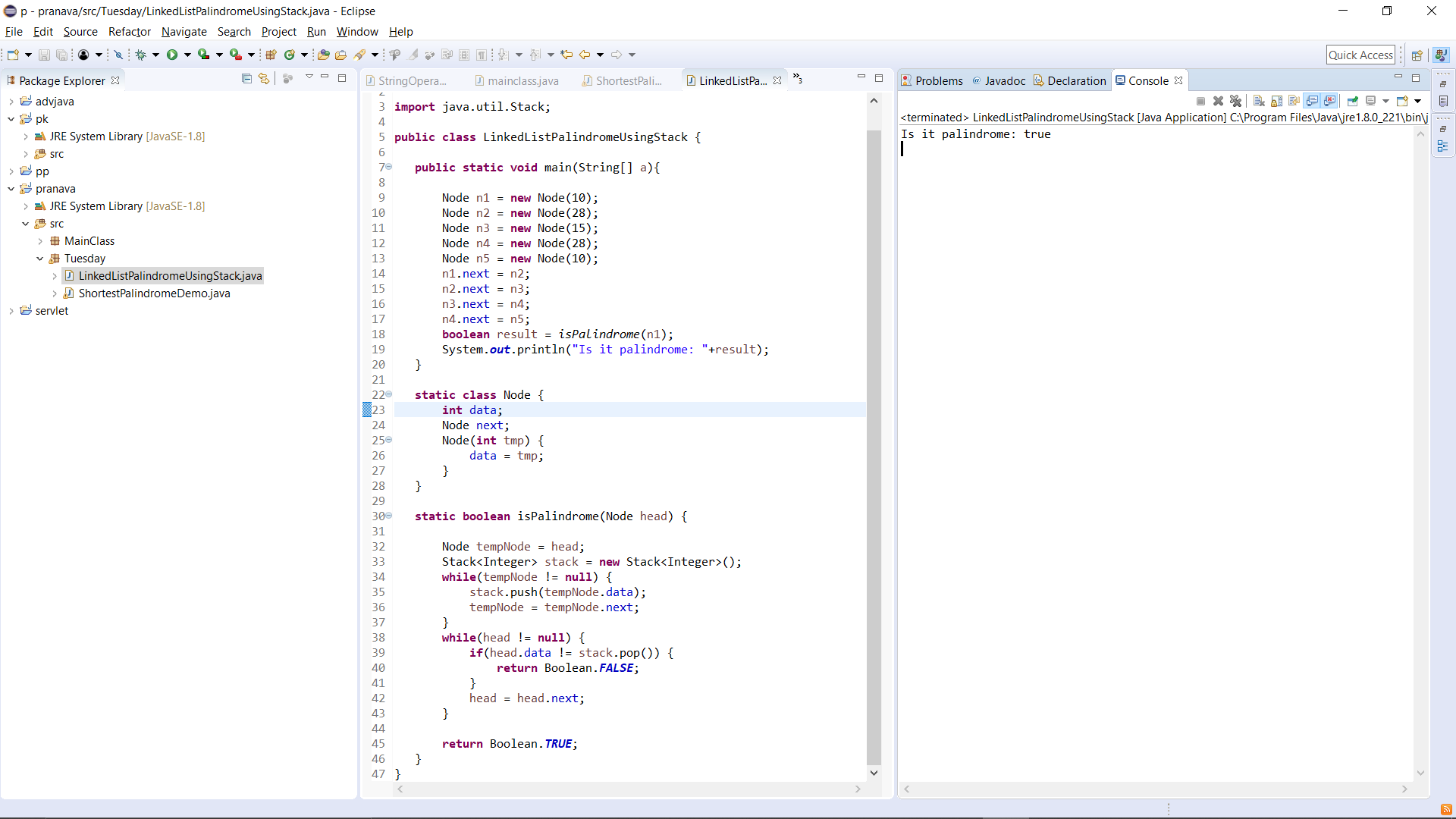
}

head = head.next;

}

**return**Boolean.***TRUE***;

}

}

