**CSA-0908 JAVA PROGRAMMING**

**EASY PROGRAMS**

1.Two Sum

import java.util.HashMap;

import java.util.Map;

public class R192211470 {

public static int[] twoSum(int[] nums, int target) {

Map<Integer, Integer> map = new HashMap<>();

for (int i = 0; i < nums.length; i++) {

int complement = target - nums[i];

if (map.containsKey(complement)) {

return new int[] { map.get(complement), i };

}

map.put(nums[i], i);

}

throw new IllegalArgumentException("No two sum solution");

}

public static void main(String[] args) {

int[] nums1 = {2, 7, 11, 15};

int target1 = 9;

int[] result1 = twoSum(nums1, target1);

System.out.println("Output: [" + result1[0] + ", " + result1[1] + "]");

int[] nums2 = {3, 2, 4};

int target2 = 6;

int[] result2 = twoSum(nums2, target2);

System.out.println("Output: [" + result2[0] + ", " + result2[1] + "]");

int[] nums3 = {3, 3};

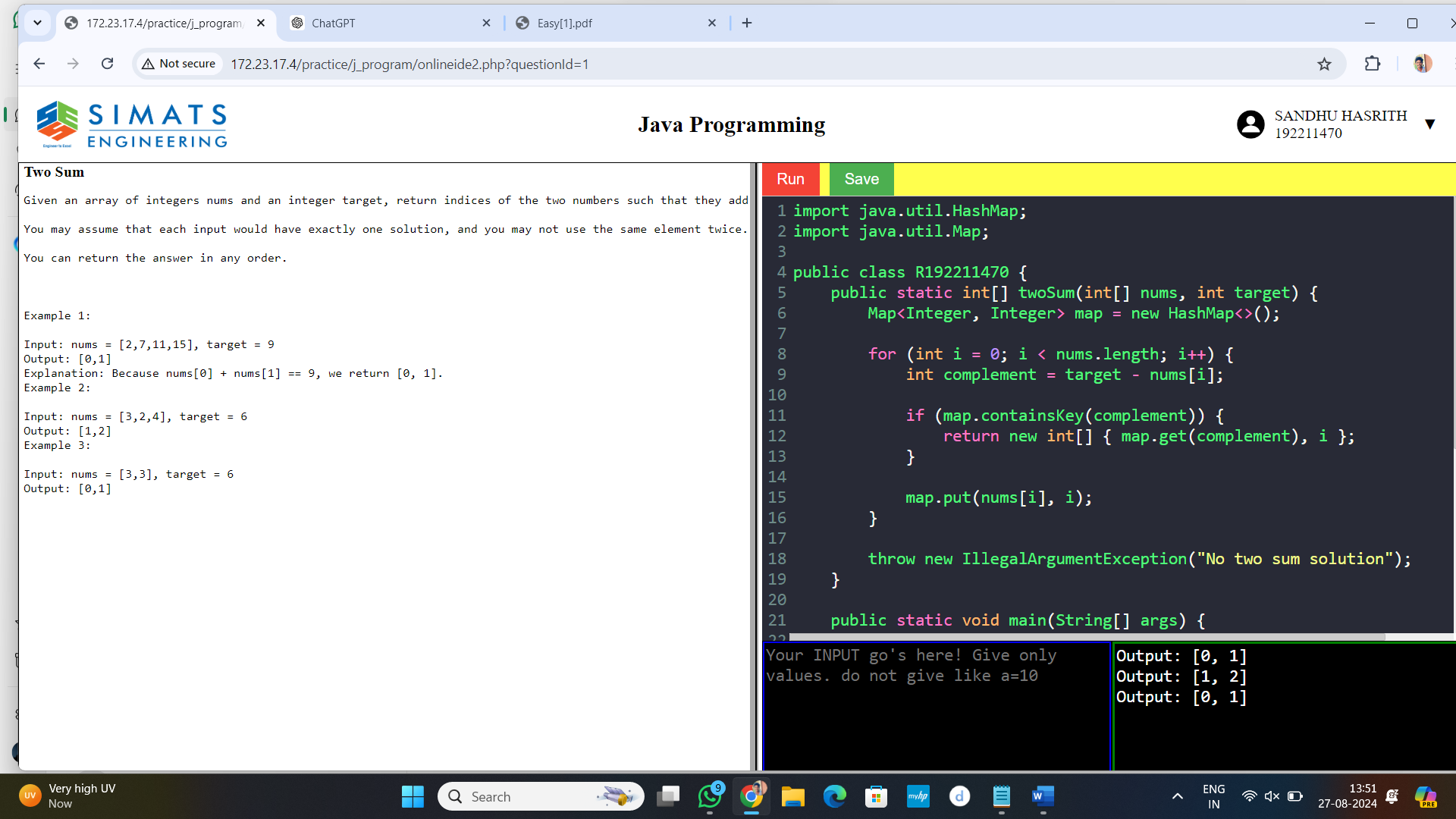
int target3 = 6;

int[] result3 = twoSum(nums3, target3);

System.out.println("Output: [" + result3[0] + ", " + result3[1] + "]");

}

}



2.Valid Palindrome

public class R192211470 {

public static boolean isPalindrome(int x) {

if (x < 0) {

return false;

}

int original = x;

int reversed = 0;

while (x != 0) {

int digit = x % 10;

reversed = reversed \* 10 + digit;

x /= 10;

}

return original == reversed;

}

public static void main(String[] args) {

int x1 = 121;

System.out.println(x1 + " is a palindrome: " + isPalindrome(x1));

int x2 = -121;

System.out.println(x2 + " is a palindrome: " + isPalindrome(x2));

int x3 = 10;

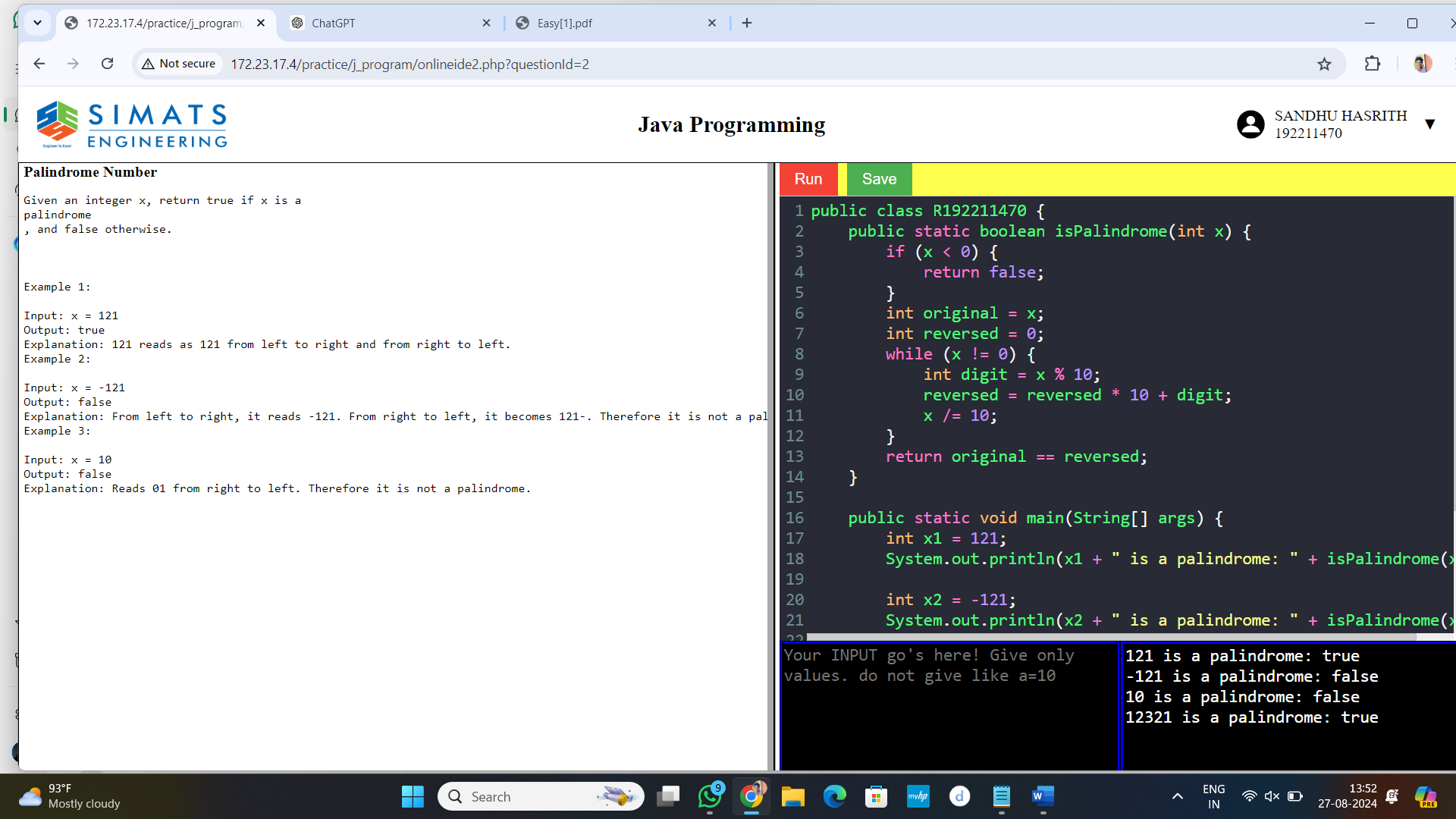
System.out.println(x3 + " is a palindrome: " + isPalindrome(x3));

int x4 = 12321;

System.out.println(x4 + " is a palindrome: " + isPalindrome(x4));

}

}



3. longest Common Prefix

public class R192211470 {

public static String longestCommonPrefix(String[] strs) {

if (strs == null || strs.length == 0) {

return "";

}

String prefix = strs[0];

for (int i = 1; i < strs.length; i++) {

while (strs[i].indexOf(prefix) != 0) {

prefix = prefix.substring(0, prefix.length() - 1);

if (prefix.isEmpty()) {

return "";

}

}

}

return prefix;

}

public static void main(String[] args) {

String[] strs1 = {"flower", "flow", "flight"};

System.out.println("Longest Common Prefix: " + longestCommonPrefix(strs1)); // Output: "fl"

String[] strs2 = {"dog", "racecar", "car"};

System.out.println("Longest Common Prefix: " + longestCommonPrefix(strs2));

String[] strs3 = {"interspecies", "interstellar", "interstate"};

System.out.println("Longest Common Prefix: " + longestCommonPrefix(strs3));

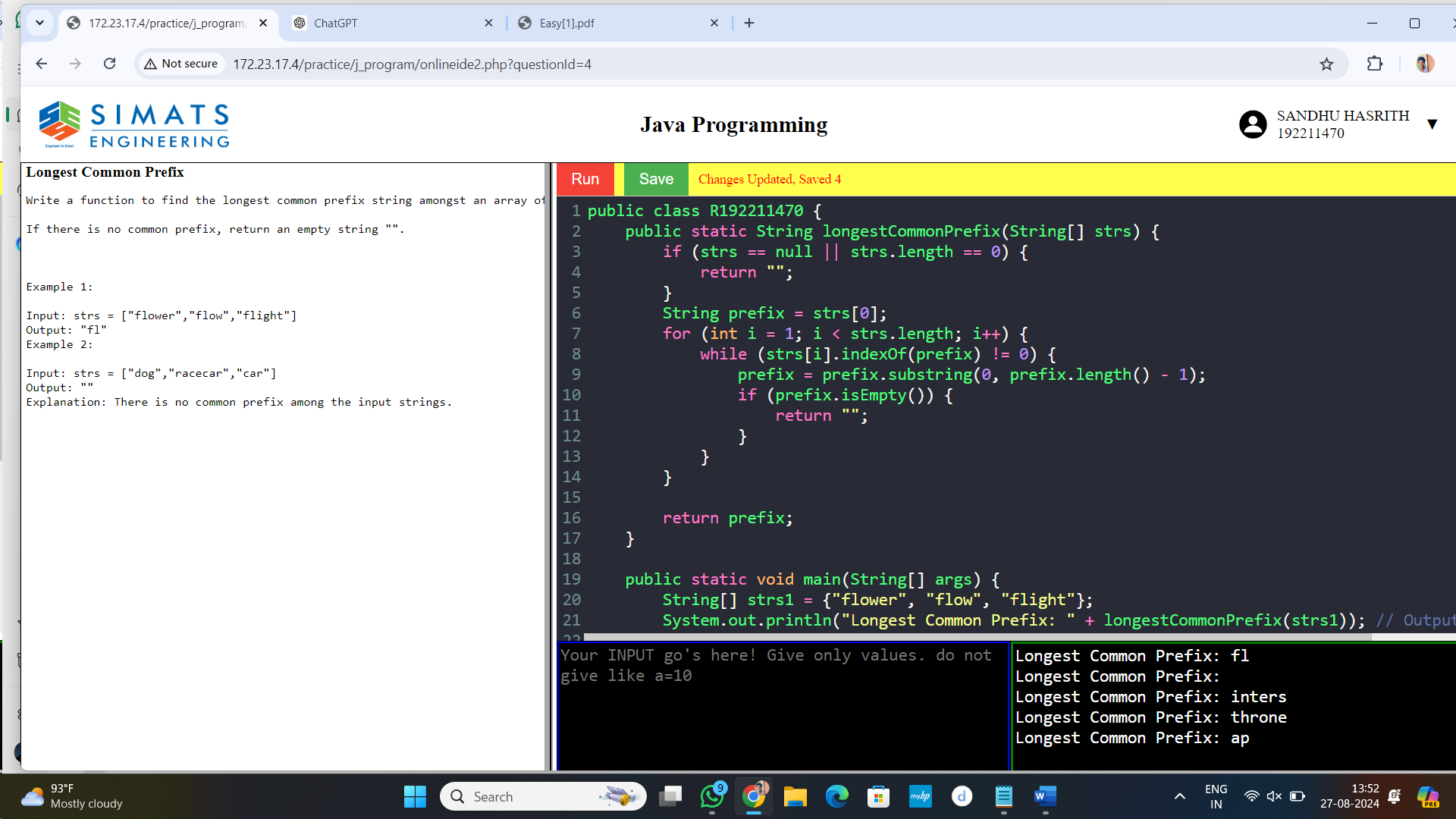
String[] strs4 = {"throne", "throne"};

System.out.println("Longest Common Prefix: " + longestCommonPrefix(strs4));

String[] strs5 = {"apple", "ape", "april"};

System.out.println("Longest Common Prefix: " + longestCommonPrefix(strs5));

}

} 

4. Merge two lists

class ListNode {

int val;

ListNode next;

ListNode(int val) { this.val = val; }

}

public class R192211470 {

public static ListNode mergeTwoLists(ListNode list1, ListNode list2) {

ListNode dummy = new ListNode(-1);

ListNode current = dummy;

while (list1 != null && list2 != null) {

if (list1.val <= list2.val) {

current.next = list1;

list1 = list1.next;

} else {

current.next = list2;

list2 = list2.next;

}

current = current.next;

}

if (list1 != null) {

current.next = list1;

} else if (list2 != null) {

current.next = list2;

}

return dummy.next;

}

public static void printList(ListNode head) {

while (head != null) {

System.out.print(head.val + " ");

head = head.next;

}

System.out.println();

}

public static void main(String[] args) {

ListNode list1 = new ListNode(1);

list1.next = new ListNode(2);

list1.next.next = new ListNode(4);

ListNode list2 = new ListNode(1);

list2.next = new ListNode(3);

list2.next.next = new ListNode(4);

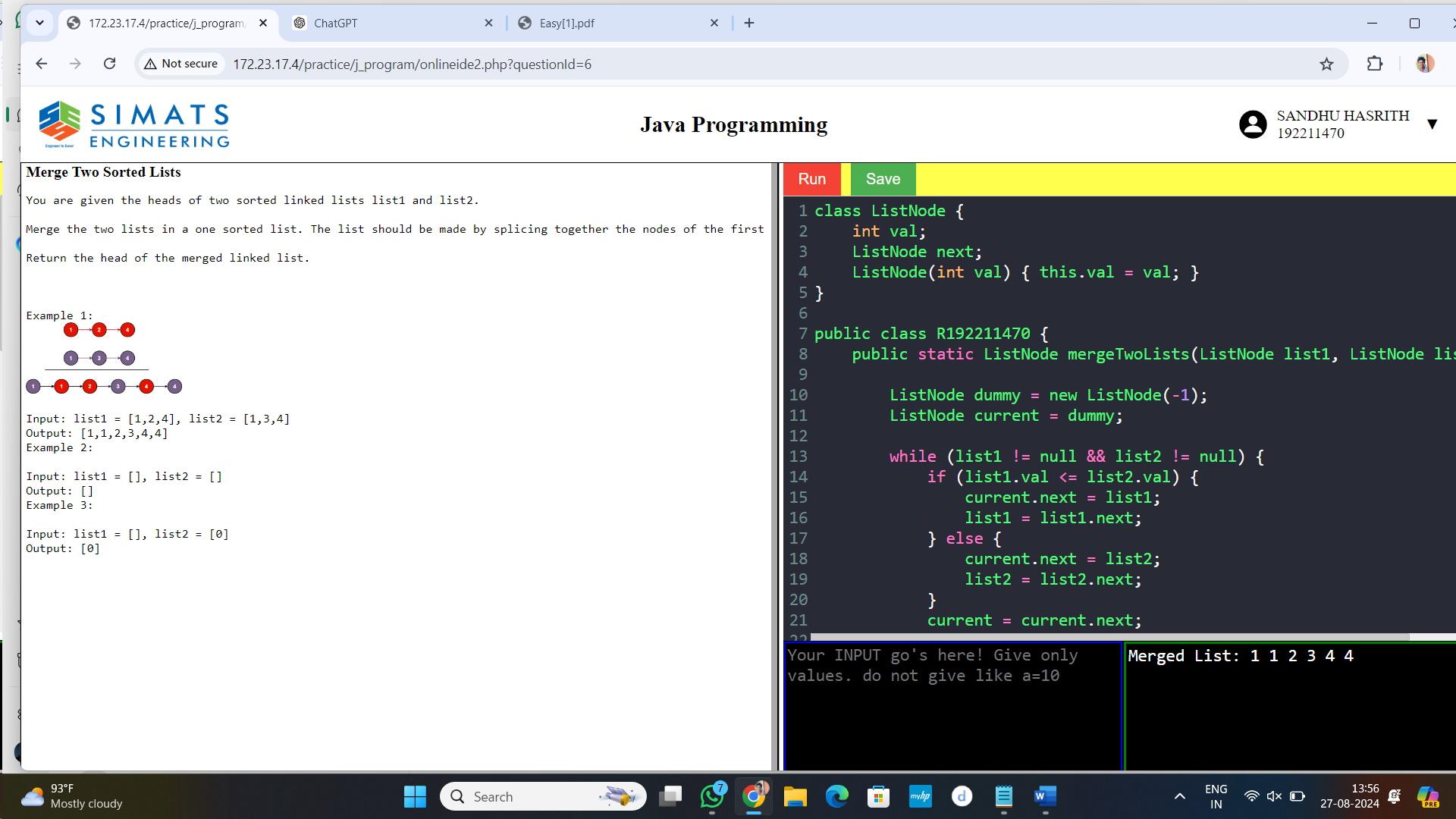
ListNode mergedList = mergeTwoLists(list1, list2);

System.out.print("Merged List: ");

printList(mergedList);

}

}



5.