**PG-DAC AUGUST 24 BATCH**

1)Write a Java program that takes a list of integers as input and returns a list of duplicate integers.

import java.util.ArrayList;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

public class DuplicateIntegers {

public static List<Integer> findDuplicates(List<Integer> inputList) {

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

List<Integer> duplicateList = new ArrayList<>();

// Count the occurrences of each integer

for (Integer num : inputList) {

if (countMap.containsKey(num)) {

countMap.put(num, countMap.get(num) + 1);

} else {

countMap.put(num, 1);

}

}

// Find the integers with more than one occurrence

for (Map.Entry<Integer, Integer> entry : countMap.entrySet()) {

if (entry.getValue() > 1) {

duplicateList.add(entry.getKey());

}

}

return duplicateList;

}

public static void main(String[] args) {

List<Integer> inputList = new ArrayList<>();

inputList.add(1);

inputList.add(2);

inputList.add(3);

inputList.add(2);

inputList.add(4);

inputList.add(5);

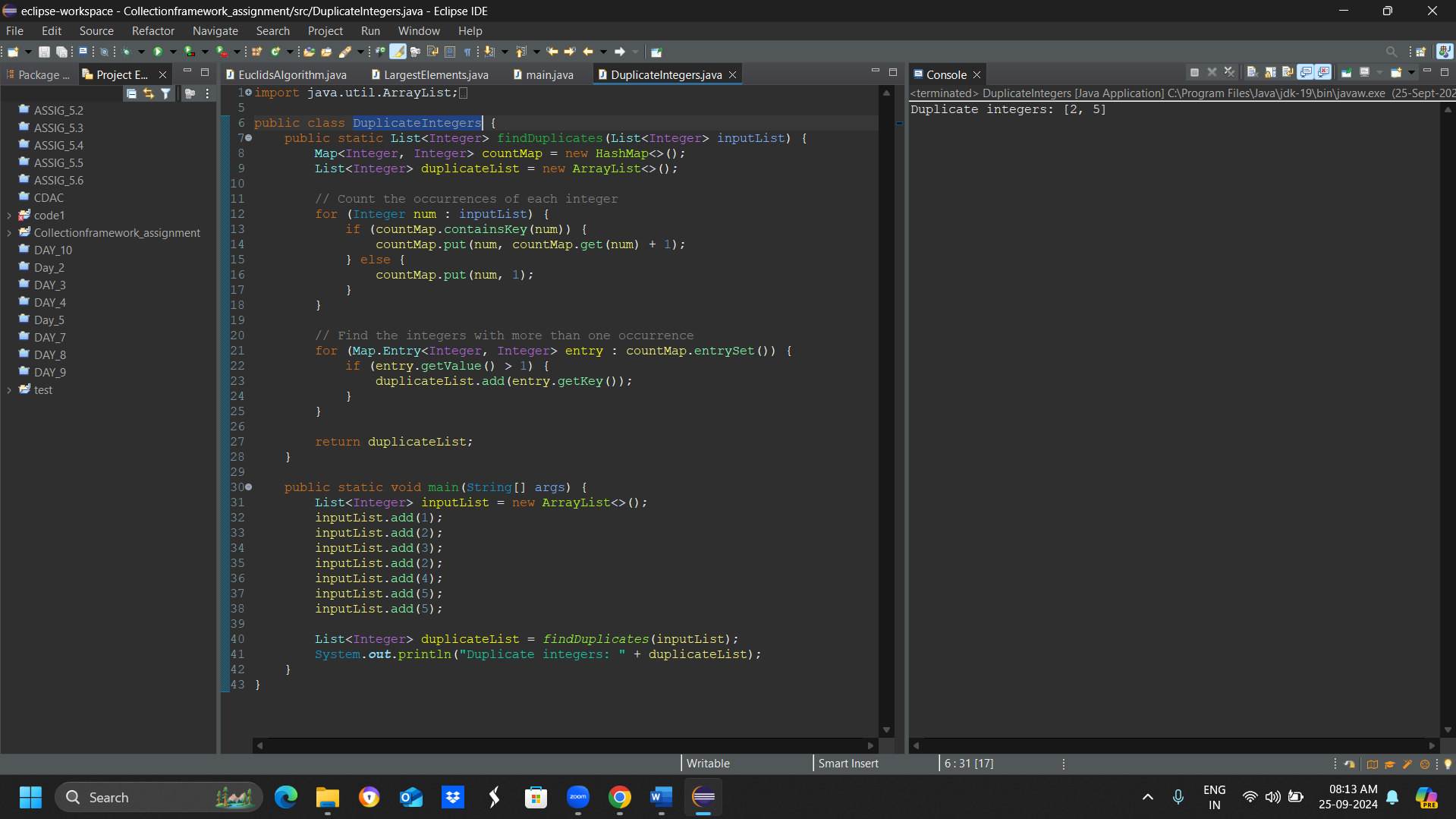
inputList.add(5);

List<Integer> duplicateList = findDuplicates(inputList);

System.out.println("Duplicate integers: " + duplicateList);

}

}



2)Create a Person class with attributes name and age. Write a Java program that sorts a list of Person objects first by age and then by name if the ages are equal.

import java.util.ArrayList;

import java.util.Collections;

import java.util.Comparator;

import java.util.List;

class Person {

private String name;

private int age;

public Person(String name, int age) {

this.name = name;

this.age = age;

}

public String getName() {

return name;

}

public int getAge() {

return age;

}

}

public class PersonSorter {

public static void main(String[] args) {

List<Person> personList = new ArrayList<>();

personList.add(new Person("John", 25));

personList.add(new Person("Alice", 30));

personList.add(new Person("Bob", 25));

personList.add(new Person("Charlie", 35));

personList.add(new Person("David", 30));

Collections.sort(personList, new Comparator<Person>() {

@Override

public int compare(Person p1, Person p2) {

if (p1.getAge() != p2.getAge()) {

return Integer.compare(p1.getAge(), p2.getAge());

} else {

return p1.getName().compareTo(p2.getName());

}

}

});

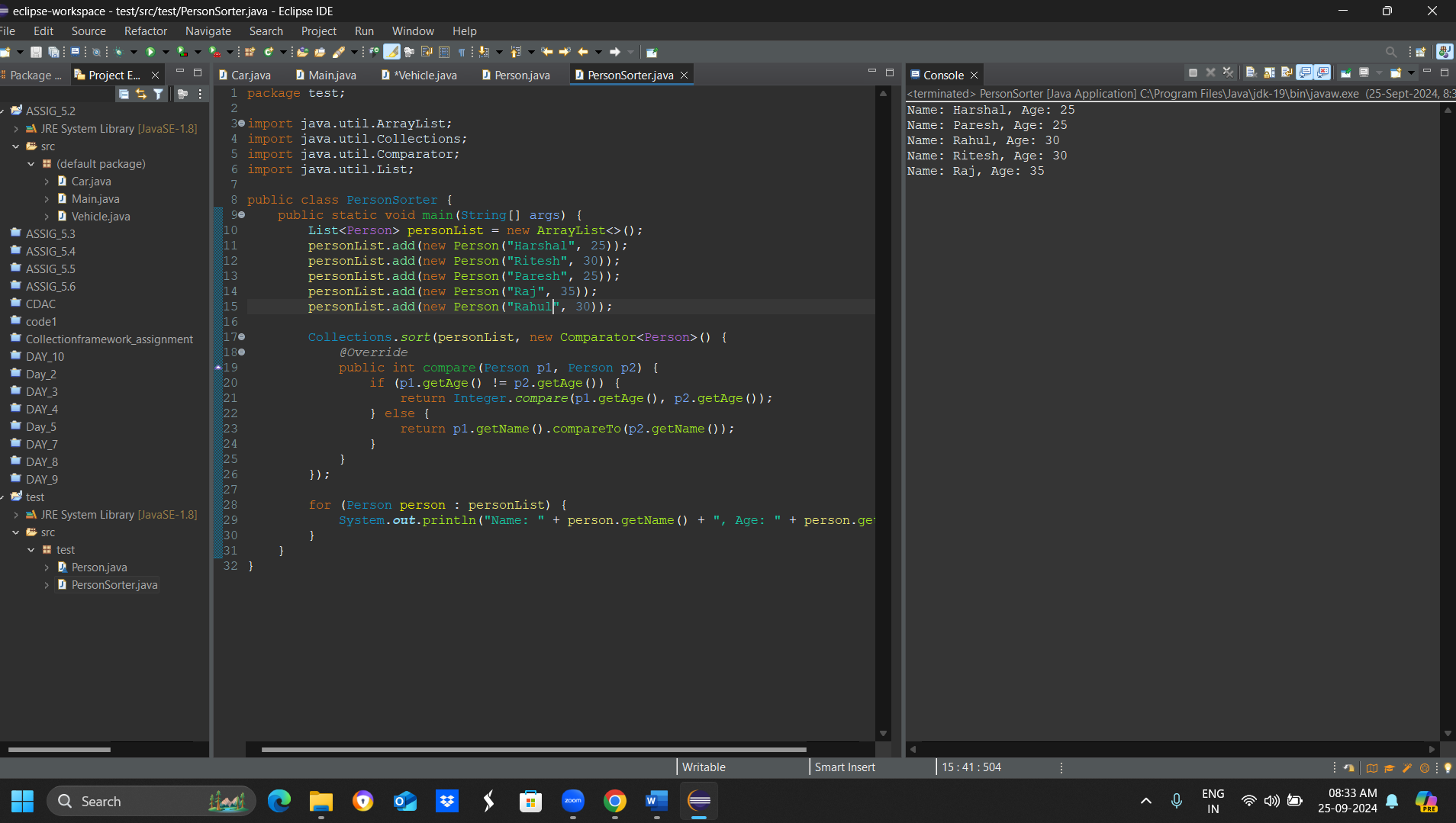
for (Person person : personList) {

System.out.println("Name: " + person.getName() + ", Age: " + person.getAge());

}

}

}



3)Write a Java program to find the first non-repeated character in a string using a HashMap.

String input = "aabbccddeffg";

Expected output = 'e';

import java.util.HashMap;

import java.util.Map;

public class FirstNonRepeatedCharacter {

public static void main(String[] args) {

String input = "aabbccddeffg";

char result = findFirstNonRepeatedCharacter(input);

System.out.println("The first non-repeated character is: " + result);

}

public static char findFirstNonRepeatedCharacter(String input) {

Map<Character, Integer> charCountMap = new HashMap<>();

// Count the occurrences of each character

for (char c : input.toCharArray()) {

charCountMap.put(c, charCountMap.getOrDefault(c, 0) + 1);

}

// Find the first non-repeated character

for (char c : input.toCharArray()) {

if (charCountMap.get(c) == 1) {

return c;

}

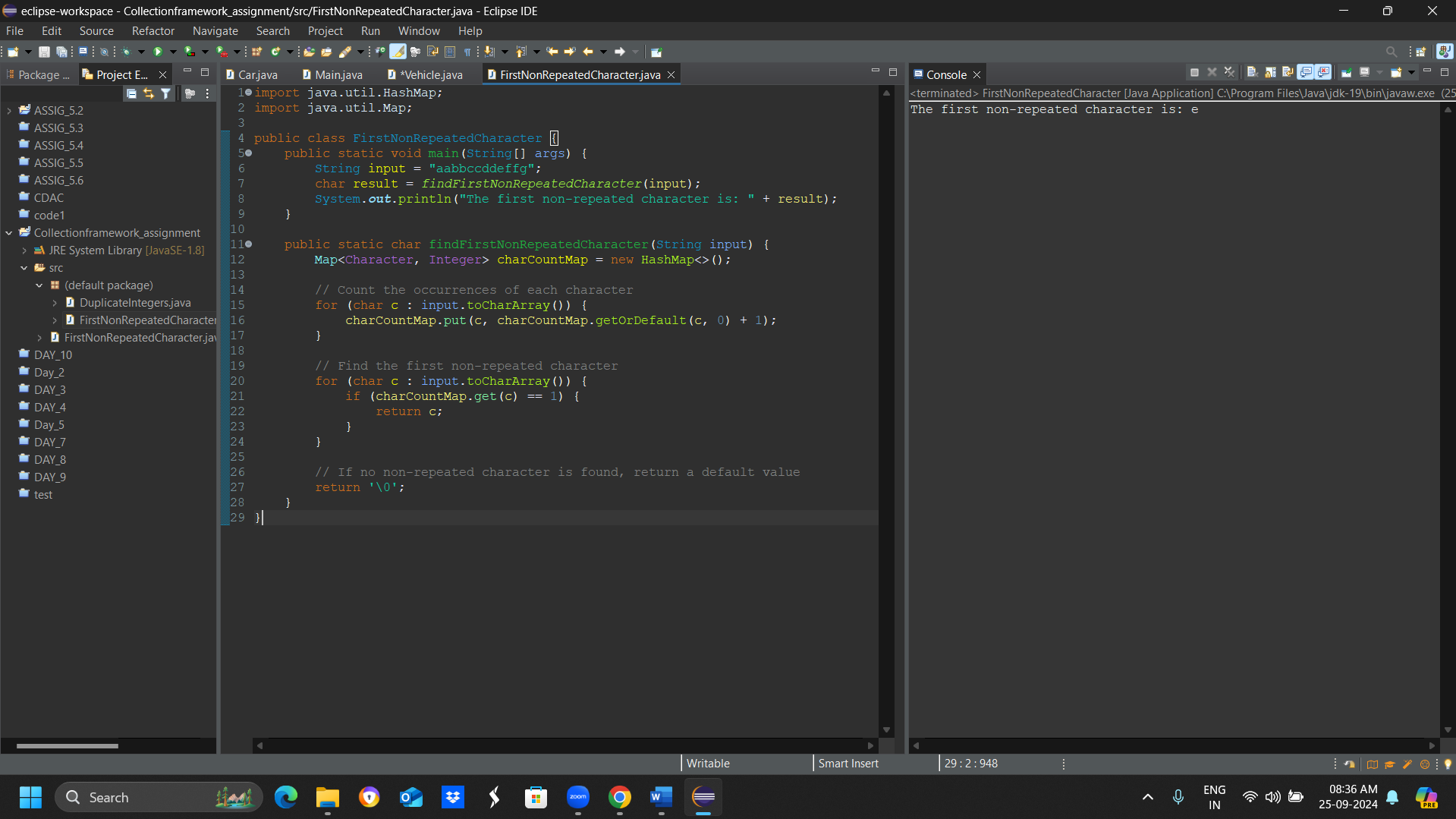
}

// If no non-repeated character is found, return a default value

return '\0';

}

}



4) Write a Java program that merges two sorted lists of integers into a single sorted list.

import java.util.ArrayList;

import java.util.Collections;

import java.util.List;

public class MergeSortedLists {

public static void main(String[] args) {

List<Integer> list1 = new ArrayList<>(List.of(1, 3, 5, 7));

List<Integer> list2 = new ArrayList<>(List.of(2, 4, 6, 8));

List<Integer> mergedList = mergeLists(list1, list2);

System.out.println("Merged list: " + mergedList);

}

public static List<Integer> mergeLists(List<Integer> list1, List<Integer> list2) {

List<Integer> mergedList = new ArrayList<>();

// Add all elements from list1 and list2 to mergedList

mergedList.addAll(list1);

mergedList.addAll(list2);

// Sort the mergedList

Collections.sort(mergedList);

return mergedList;

}

}

