ASSIGNMENT – 2.2

Subject: CSW2 (CSE 2141)

Name: Arpit Kumar Mohanty

Registration Number: 2341013237

Section: 23412G1

Branch: CSE

Q1. Create a generic class Pair with private member variables key and value. The class should include a parameterized constructor and provide getter and setter methods for these attributes. In the main class, create and add objects of the Pair class, then retrieve and print the key-value pairs.

Solution along with Output:

Q2. Write a Java program that includes a User class and an ArrayListUser class. The User class should have private fields for name and age, along with a parameterized constructor and getter/setter methods for these attributes. The ArrayListUser class should create an ArrayList of User objects. After adding user objects, it should retrieve and print their name and age. Additionally, the program should sort the users based on age and print the updated list of users using getter methods.

Solution:

Q1.java | Q2.java | Q3.java | Q4.java | Q5.java | Q6.java | Q7.java | Q8.java

Q2.java > Q2 > main(String[])

```java
import java.util.*;
class User {
    private String name;
    private int age;
    public User(String name, int age) {
        this.name = name;
        this.age = age;
    }
    public String getName() { return name; }
    public int getAge() { return age; }
    public void setName(String name) { this.name = name; }
    public void setAge(int age) { this.age = age; }

    public void display() {
        System.out.println("Name: " + name + " | Age: " + age);
    }
}
public class Q2 {
    Run | Debug
    public static void main(String[] args) {
        // Creating an ArrayList of User objects
        ArrayList<User> arl = new ArrayList<>();

        arl.add(new User(name:"Arpit Kumar", age:20));
        arl.add(new User(name:"Manshik Kumar", age:22));
        arl.add(new User(name:"Sourav Sahoo", age:19));

        System.out.println(x:"Users before sorting:");
        for (User user : arl) {
            user.display();
        }

        // Sorting users by age
        Collections.sort(arl, Comparator.comparingInt(User::getAge));

        System.out.println(x:"\nUsers after sorting by age:");
        for (User user : arl) {
            user.display();
        }
    }
}
```

Output:

```
[Running] cd "a:\Programs\HTML & CSS (from Sems)\4th Semester\CSW-
Users before sorting:
Name: Arpit Kumar | Age: 20
Name: Manshik Kumar | Age: 22
Name: Sourav Sahoo | Age: 19

Users after sorting by age:
Name: Sourav Sahoo | Age: 19
Name: Arpit Kumar | Age: 20
Name: Manshik Kumar | Age: 22

[Done] exited with code=0 in 1.124 seconds
```

Q3. Write a Java program that includes a Car class and a CarApp class. The Car class should have private fields: modelNo (int), name (String), and stock (int). Define a parameterized constructor and override the compareTo method as public int compareTo(Car car) to enable sorting of cars based on the total stock available. In the CarApp class, create an ArrayList of Car objects, sort them, and print the updated sorted list. Example of a sorted list of Car objects:

2013 Creta 10

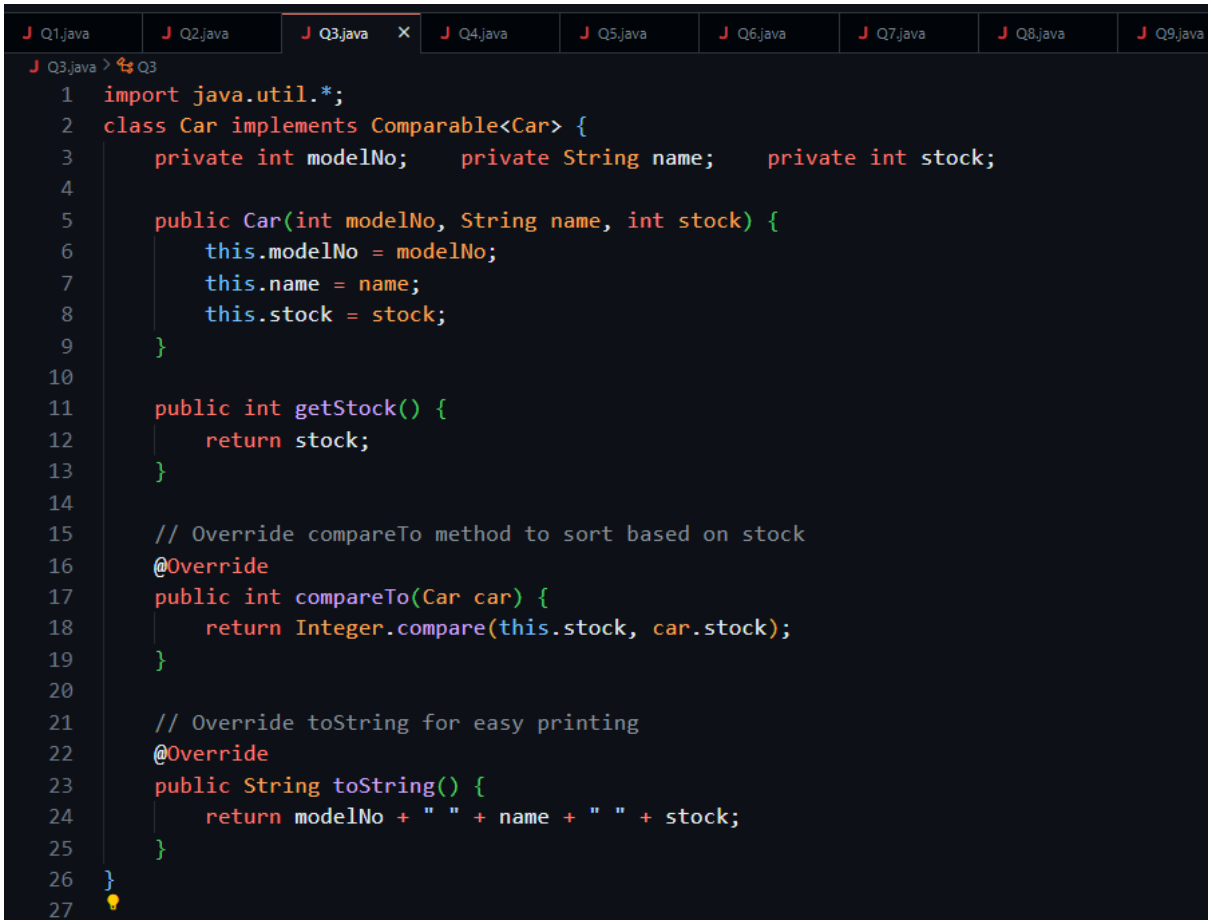
2020 MG 13

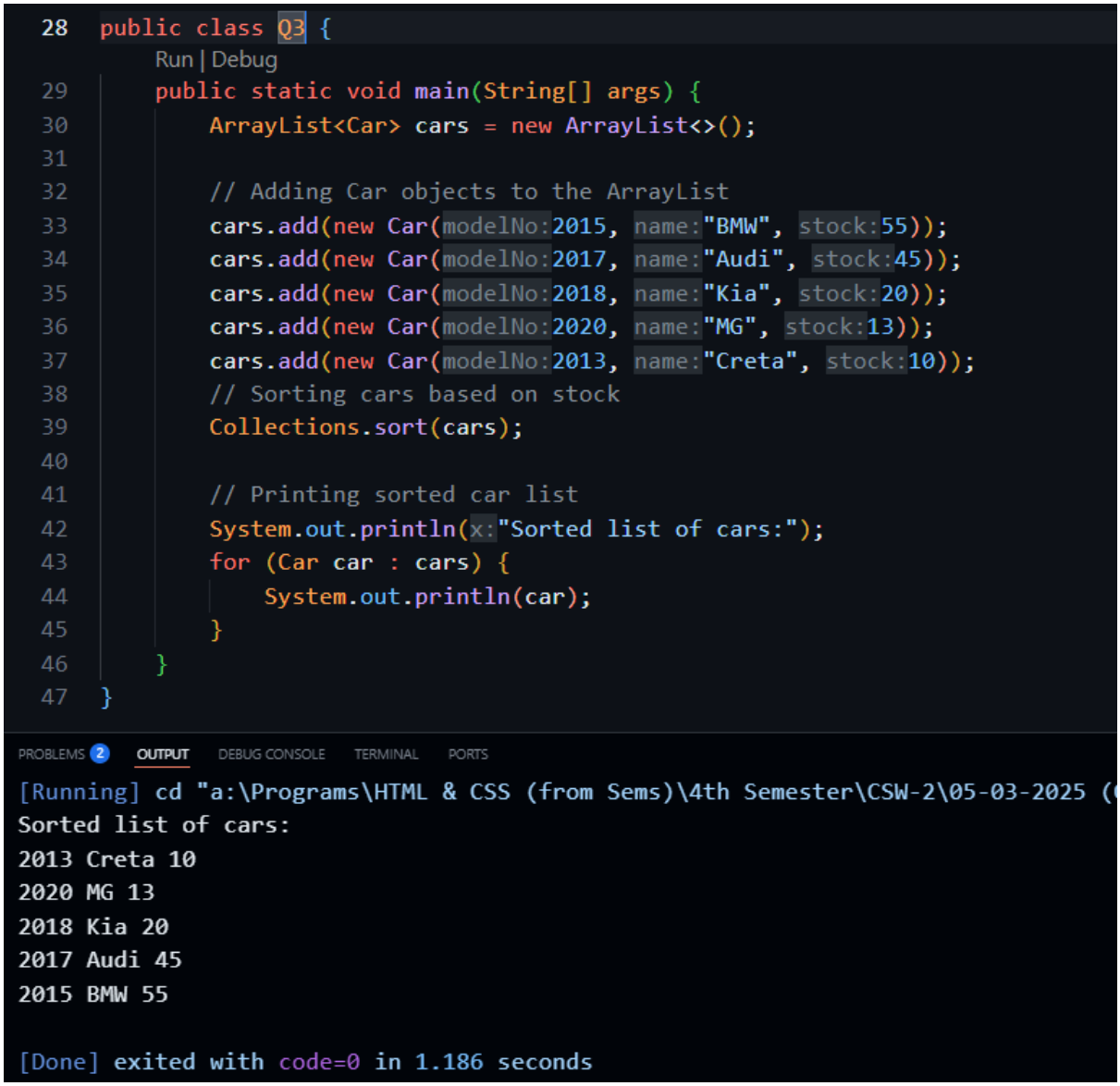
2018 Kia 20

2017 Audi 45

2015 BMW 55

Solution along with Output:


Q1.java
Q2.java
Q3.java
Q4.java
Q5.java
Q6.java
Q7.java
Q8.java
Q9.java
Q3.java > Q3
import java.util.*;
class Car implements Comparable<Car> {
private int modelNo; private String name; private int stock;
public Car(int modelNo, String name, int stock) {
this.modelNo = modelNo;
this.name = name;
this.stock = stock;
}
public int getStock() {
return stock;
}
// Override compareTo method to sort based on stock
@Override
public int compareTo(Car car) {
return Integer.compare(this.stock, car.stock);
}
// Override toString for easy printing
@Override
public String toString() {
return modelNo + " " + name + " " + stock;
}
}

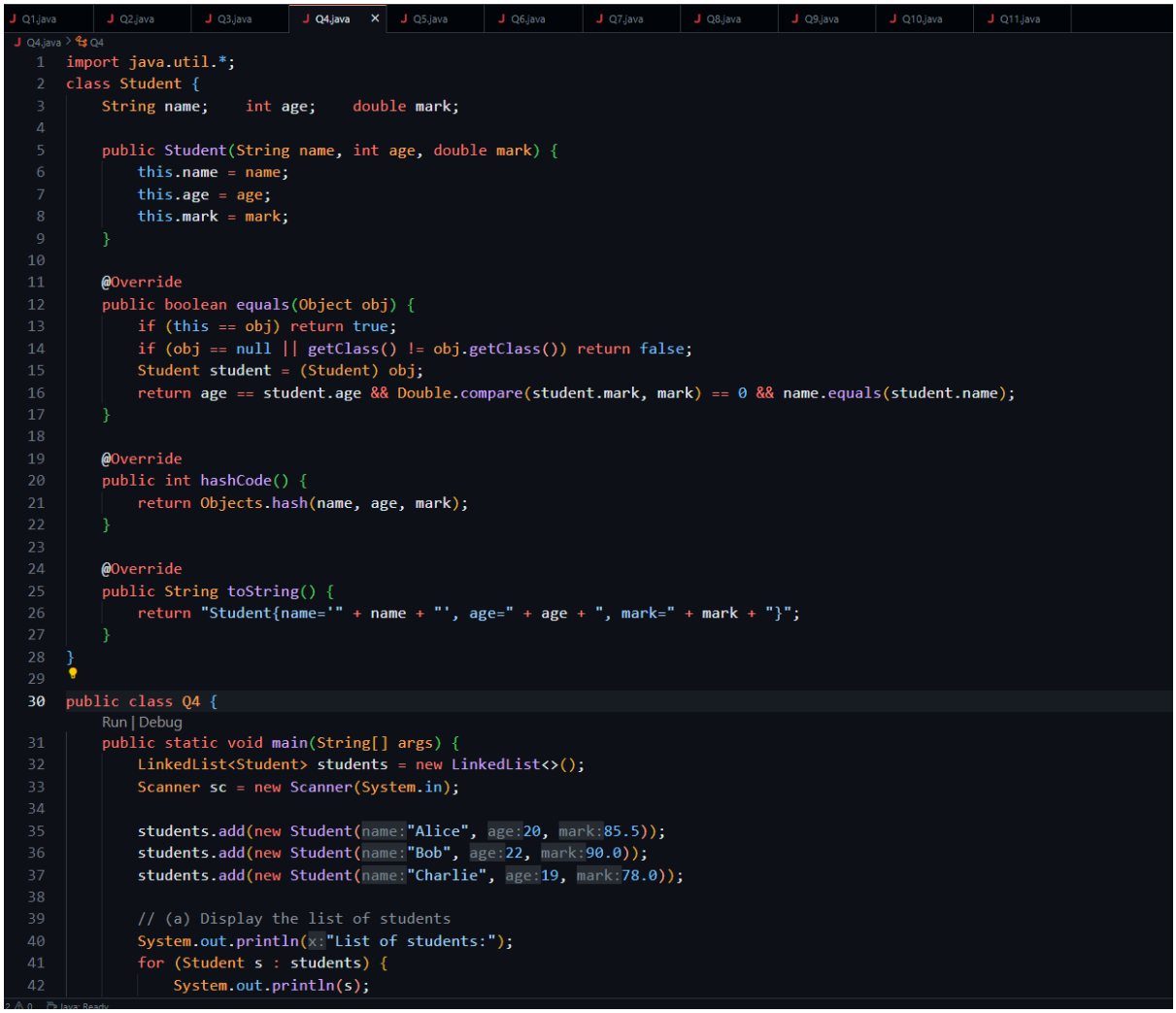


```java
public class Q3 {
    Run | Debug
    public static void main(String[] args) {
        ArrayList<Car> cars = new ArrayList<>();

        // Adding Car objects to the ArrayList
        cars.add(new Car(modelNo:2015, name:"BMW", stock:55));
        cars.add(new Car(modelNo:2017, name:"Audi", stock:45));
        cars.add(new Car(modelNo:2018, name:"Kia", stock:20));
        cars.add(new Car(modelNo:2020, name:"MG", stock:13));
        cars.add(new Car(modelNo:2013, name:"Creta", stock:10));
        // Sorting cars based on stock
        Collections.sort(cars);

        // Printing sorted car list
        System.out.println(x:"Sorted list of cars:");
        for (Car car : cars) {
            System.out.println(car);
        }
    }
}
```

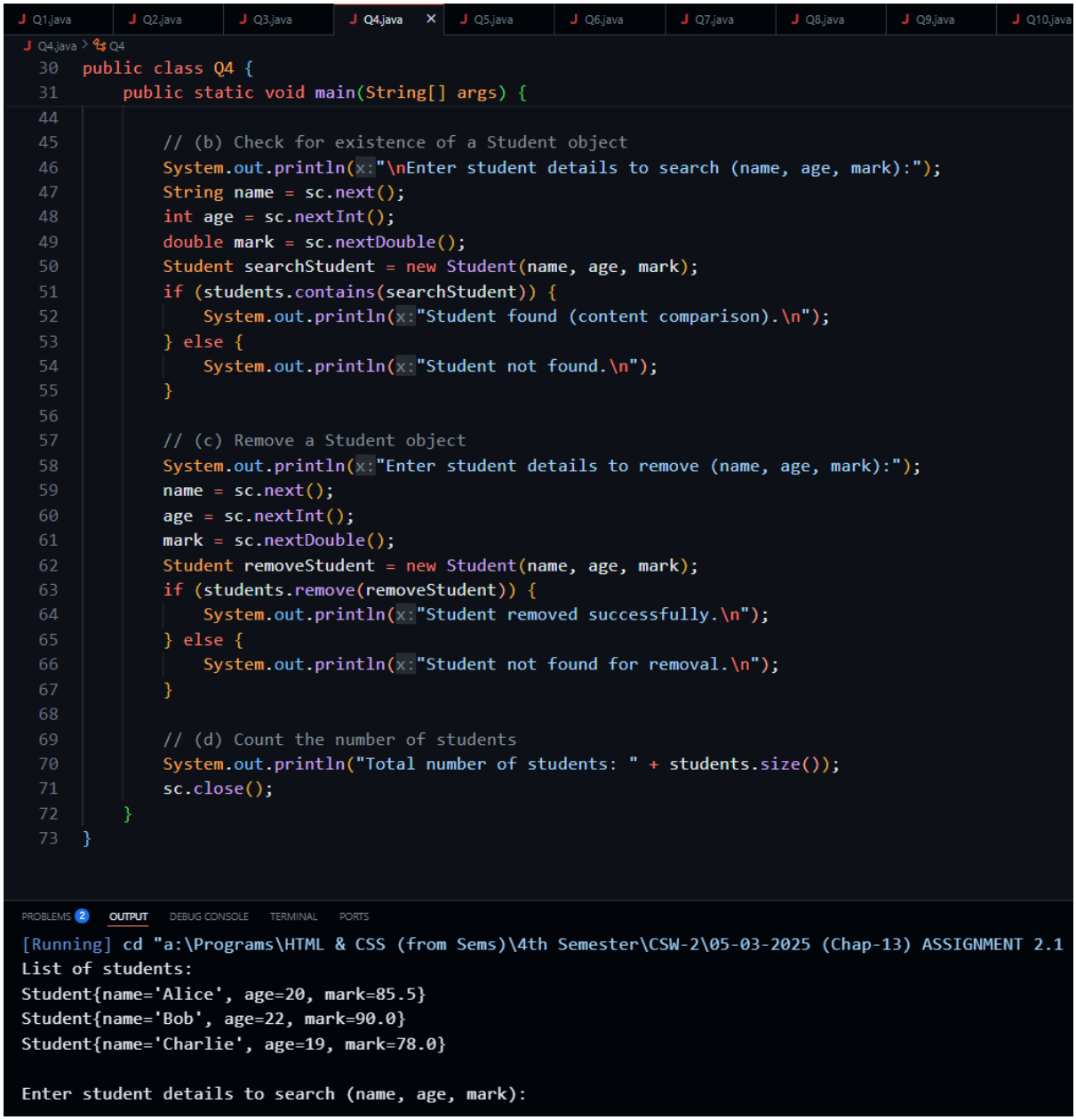
```
PROBLEMS 2  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
[Running] cd "a:\Programs\HTML & CSS (from Sems)\4th Semester\CSW-2\05-03-2025 (
Sorted list of cars:
2013 Creta 10
2020 MG 13
2018 Kia 20
2017 Audi 45
2015 BMW 55

[Done] exited with code=0 in 1.186 seconds
```

Q4. Create a Student class with member variables name, age, and mark, along with the necessary getter and setter methods. Implement a LinkedList of Student objects and perform the following operations: (a) Display the list of students. (b) Prompt the user to enter a Student object and check its existence in the list. Specify whether the search is based on reference comparison or content comparison using the contains method. (c) Remove a specified Student object from the list. (d) Count the number of Student objects present in the list. (e) Override the equals method to compare two Student objects based on their values rather than references

Solution along with Output:



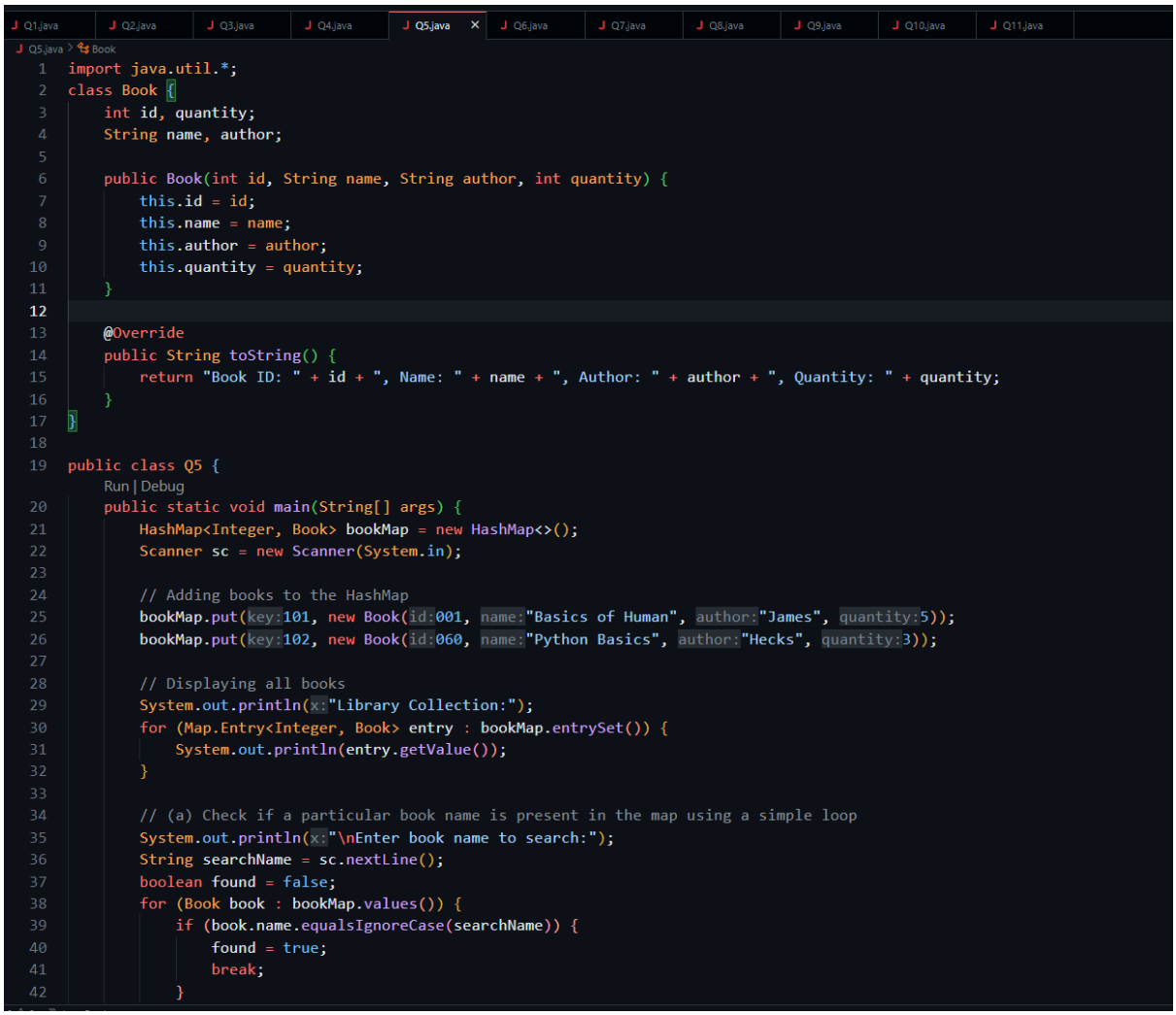
```java
import java.util.*;
class Student {
    String name;    int age;    double mark;

    public Student(String name, int age, double mark) {
        this.name = name;
        this.age = age;
        this.mark = mark;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (obj == null || getClass() != obj.getClass()) return false;
        Student student = (Student) obj;
        return age == student.age && Double.compare(student.mark, mark) == 0 && name.equals(student.name);
    }

    @Override
    public int hashCode() {
        return Objects.hash(name, age, mark);
    }

    @Override
    public String toString() {
        return "Student{name='" + name + "', age=" + age + ", mark=" + mark + "}";
    }
}

public class Q4 {
    Run | Debug
    public static void main(String[] args) {
        LinkedList<Student> students = new LinkedList<>();
        Scanner sc = new Scanner(System.in);

        students.add(new Student(name:"Alice", age:20, mark:85.5));
        students.add(new Student(name:"Bob", age:22, mark:90.0));
        students.add(new Student(name:"Charlie", age:19, mark:78.0));

        // (a) Display the list of students
        System.out.println(x:"List of students:");
        for (Student s : students) {
            System.out.println(s);
```



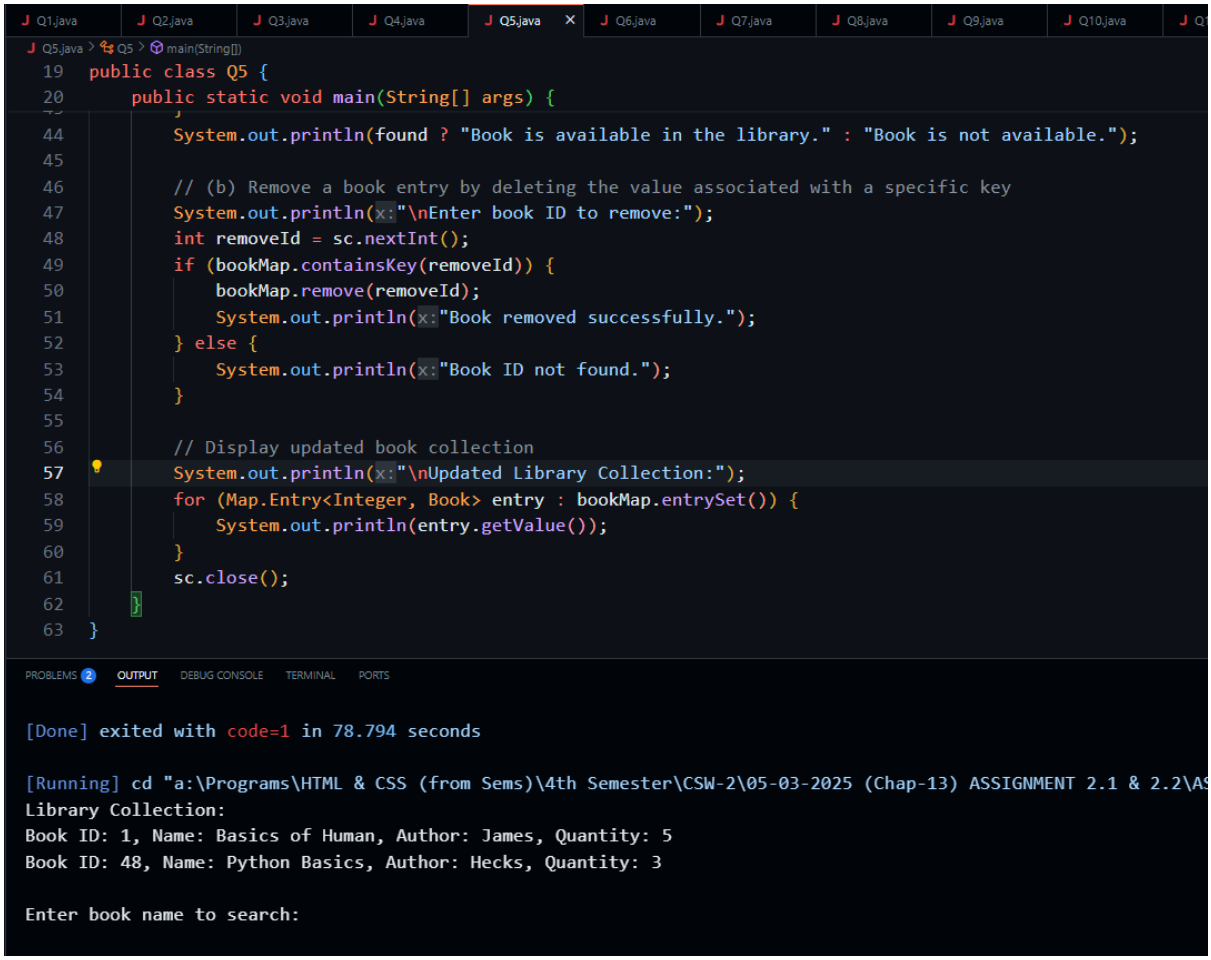
Q5. Create a Book class with member variables id, name, author, and quantity to store details of each issued book. The Book class should include a parameterized constructor. Design a Library class that creates a HashMap of books, where the key is an Integer (representing the book ID) and the value is a Book object. Instantiate at least two Book objects and display the collection of books stored in the HashMap. Use appropriate methods of the HashMap class to perform the following operations: (a) Check if a particular book name is present on the map. (b) Remove a book entry by deleting the value associated with a specific

key.

Solution:



```java
import java.util.*;
class Book {
    int id, quantity;
    String name, author;

    public Book(int id, String name, String author, int quantity) {
        this.id = id;
        this.name = name;
        this.author = author;
        this.quantity = quantity;
    }

    @Override
    public String toString() {
        return "Book ID: " + id + ", Name: " + name + ", Author: " + author + ", Quantity: " + quantity;
    }
}

public class Q5 {
    Run | Debug
    public static void main(String[] args) {
        HashMap<Integer, Book> bookMap = new HashMap<>();
        Scanner sc = new Scanner(System.in);

        // Adding books to the HashMap
        bookMap.put(key:101, new Book(id:001, name:"Basics of Human", author:"James", quantity:5));
        bookMap.put(key:102, new Book(id:060, name:"Python Basics", author:"Hecks", quantity:3));

        // Displaying all books
        System.out.println(x:"Library Collection:");
        for (Map.Entry<Integer, Book> entry : bookMap.entrySet()) {
            System.out.println(entry.getValue());
        }

        // (a) Check if a particular book name is present in the map using a simple loop
        System.out.println(x:"\nEnter book name to search:");
        String searchName = sc.nextLine();
        boolean found = false;
        for (Book book : bookMap.values()) {
            if (book.name.equalsIgnoreCase(searchName)) {
                found = true;
                break;
            }
```



Q6. Write a program to create a TreeSet of Integer type and perform the following operations: (a) Display the elements of the TreeSet. (b) Prompt the user to enter a number and check whether the number is present in the TreeSet. (c) Remove a specified element from the TreeSet

Solution:

```java
import java.util.*;
public class Q6 {
    Run | Debug
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        TreeSet<Integer> treeSet = new TreeSet<>();

        for (int i = 10; i <= 50; i += 10) {            treeSet.add(i);        }

        // (a) Display the elements of the TreeSet
        System.out.println("TreeSet elements: " + treeSet);

        // (b) Check if a user-input number is present in the TreeSet
        System.out.print("Enter a number to check: ");
        int numberToCheck = sc.nextInt();
        System.out.println(numberToCheck + (treeSet.contains(numberToCheck) ? " is present in the TreeSet." : " is not present in the TreeSet."));

        // (c) Remove a specified element from the TreeSet
        System.out.print("Enter a number to remove: ");
        int numberToRemove = sc.nextInt();
        System.out.println(numberToRemove + (treeSet.remove(numberToRemove) ? " has been removed from the TreeSet." : " was not found in the TreeSet."));

        // Display updated TreeSet
        System.out.println("Updated TreeSet: " + treeSet);
        sc.close();
    }
}
```

```
[Done] exited with code=1 in 45.976 seconds

[Running] cd "a:\Programs\HTML & CSS (from Sems)\4th Semester\CSW-2\05-03-2025 (Chap-13) ASSIGNMENT 2.1 & 2.2\ASSIGNMENT 2.2\" && javac Q6.java && java Q6
TreeSet elements: [10, 20, 30, 40, 50]
Enter a number to check:
```

Q7. Write a Java program that includes a class Address with member variables plotNo, at, and post. The class should define a parameterized constructor to initialize these attributes. Create a TreeMap, where the key is the name of a person (String), and the value is an Address object. Insert the required key-value pairs into the TreeMap and use an Iterator to display the entries.

Solution:

```java
import java.util.*;
class Address {
    int plotNo;    String at;    String post;

    public Address(int plotNo, String at, String post) {
        this.plotNo = plotNo;
        this.at = at;
        this.post = post;
    }

    @Override
    public String toString() {
        return "Plot No: " + plotNo + ", At: " + at + ", Post: " + post;
    }
}

public class Q7 {
    Run | Debug
    public static void main(String[] args) {
        TreeMap<String, Address> addressMap = new TreeMap<>();

        // Adding entries to the TreeMap
        addressMap.put(key:"Arpit", new Address(plotNo:159, at:"ABC Lane", post:"Delhi"));
        addressMap.put(key:"Bhuvan", new Address(plotNo:202, at:"BCD Lane", post:"BBSR"));
        addressMap.put(key:"Chinmayee", new Address(plotNo:893, at:"EDC Lane", post:"Puri"));

        // Using an iterator to display the entries
        Iterator<Map.Entry<String, Address>> iterator = addressMap.entrySet().iterator();
        while (iterator.hasNext()) {
            Map.Entry<String, Address> entry = iterator.next();
            System.out.println("Name: " + entry.getKey() + ", Address: " + entry.getValue());
        }
    }
}
```

```
[Running] cd "a:\Programs\HTML & CSS (from Sems)\4th Semester\CSW-2\05-03-2025 (Chap-13) ASSIGNMENT 2.1 & 2.2\ASSIGNMENT 2
Name: Arpit, Address: Plot No: 159, At: ABC Lane, Post: Delhi
Name: Bhuvan, Address: Plot No: 202, At: BCD Lane, Post: BBSR
Name: Chinmayee, Address: Plot No: 893, At: EDC Lane, Post: Puri
```

Q8. Write a Java program to determine whether two given strings are anagrams. An anagram is a word or phrase formed by rearranging the letters of another word or phrase. Declare two strings, str1 and str2, and initialize them with values. Create a HashMap to store the character frequencies of one string. Use the methods containsKey(), put(), and get() to compare both strings and verify if they are anagrams.

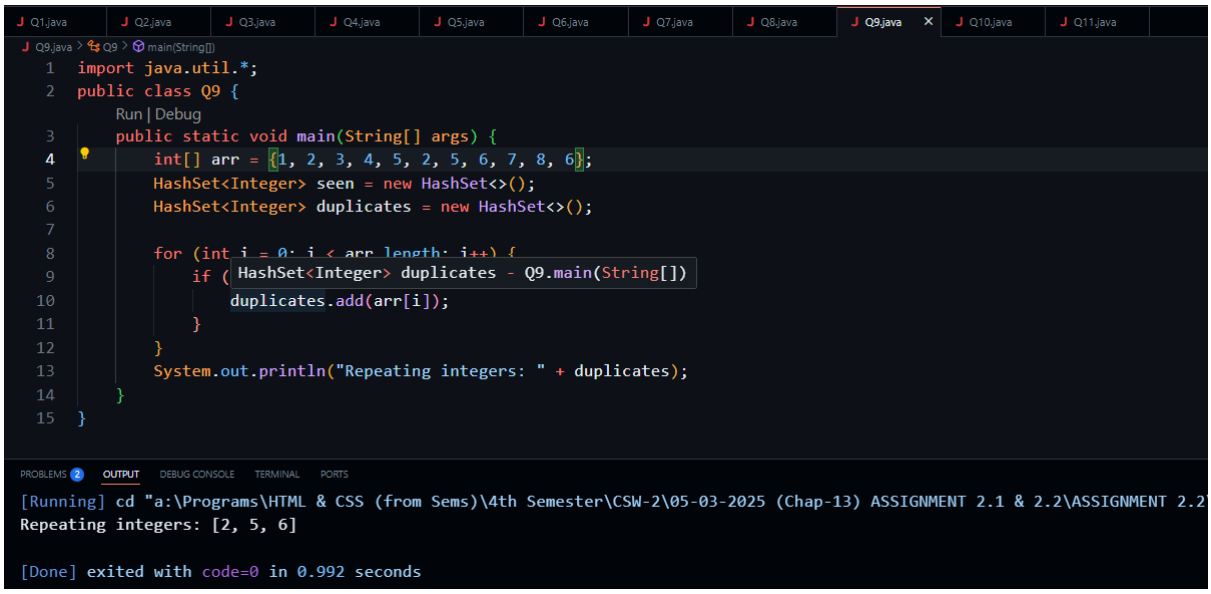
Solution:

```java
import java.util.*;
public class Q8 {
    public static boolean areAnagrams(String str1, String str2) {
        if (str1.length() != str2.length()) return false;

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

        for (char c : str1.toCharArray())
            charCountMap.put(c, charCountMap.getOrDefault(c, defaultValue:0) + 1);

        for (char c : str2.toCharArray()) {
            if (!charCountMap.containsKey(c) || charCountMap.get(c) == 0) return false;
            charCountMap.put(c, charCountMap.get(c) - 1);
        }
        return true;
    }

    Run | Debug
    public static void main(String[] args) {
        String str1 = "listen", str2 = "silent";
        System.out.println(str1 + " and " + str2 + (areAnagrams(str1, str2) ? " are " : " are not ") + "anagrams.");
    }
}
```

```
[Running] cd "a:\Programs\HTML & CSS (from Sems)\4th Semester\CSW-2\05-03-2025 (Chap-13) ASSIGNMENT 2.1 & 2.2\ASSIGNMENT 2.2\" &
listen and silent are anagrams.
```

Q9. Given an array of integers, write a Java program to identify and print the repeating integers using a HashSet.

Solution:



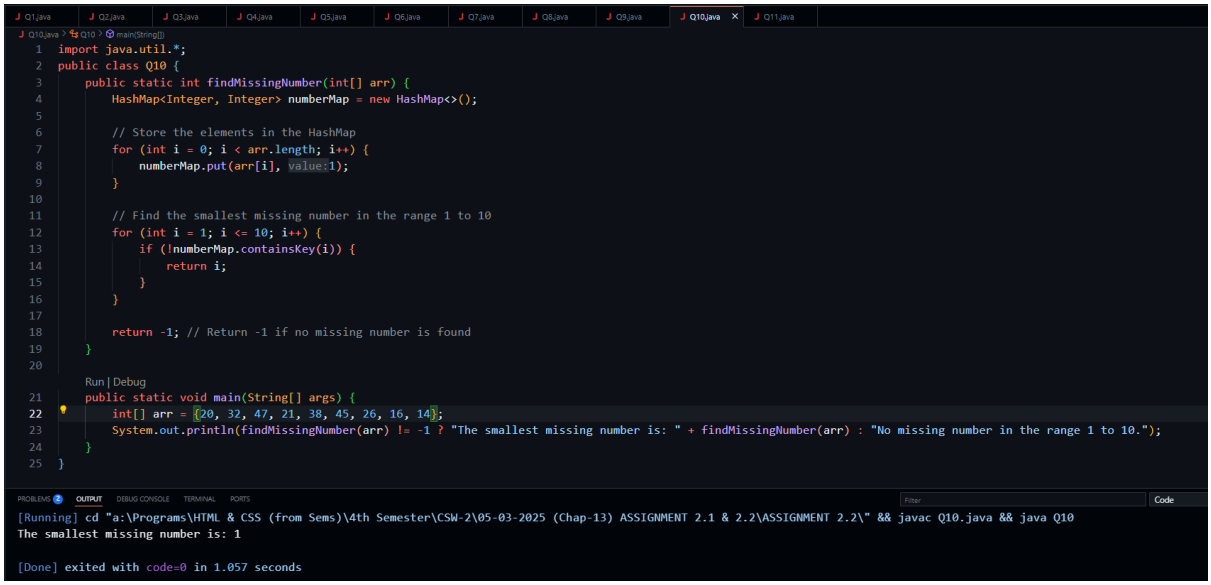
```java
import java.util.*;
public class Q9 {
    Run | Debug
    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, 5, 2, 5, 6, 7, 8, 6};
        HashSet<Integer> seen = new HashSet<>();
        HashSet<Integer> duplicates = new HashSet<>();

        for (int i = 0; i < arr.length; i++) {
            if (HashSet<Integer> duplicates - Q9.main(String[])
                duplicates.add(arr[i]);
            }
        }
        System.out.println("Repeating integers: " + duplicates);
    }
}
```

```
[Running] cd "a:\Programs\HTML & CSS (from Sems)\4th Semester\CSW-2\05-03-2025 (Chap-13) ASSIGNMENT 2.1 & 2.2\ASSIGNMENT 2.2
Repeating integers: [2, 5, 6]

[Done] exited with code=0 in 0.992 seconds
```

Q10. Given an unsorted array of integers ranging from 1 to 10, write a program to find the smallest positive number missing in the array. Use a HashMap to keep track of the elements and identify the missing

number.

Solution:



```java
import java.util.*;
public class Q10 {
    public static int findMissingNumber(int[] arr) {
        HashMap<Integer, Integer> numberMap = new HashMap<>();

        // Store the elements in the HashMap
        for (int i = 0; i < arr.length; i++) {
            numberMap.put(arr[i], value:1);
        }

        // Find the smallest missing number in the range 1 to 10
        for (int i = 1; i <= 10; i++) {
            if (!numberMap.containsKey(i)) {
                return i;
            }
        }

        return -1; // Return -1 if no missing number is found
    }

    Run | Debug
    public static void main(String[] args) {
        int[] arr = {20, 32, 47, 21, 38, 45, 26, 16, 14};
        System.out.println(findMissingNumber(arr) != -1 ? "The smallest missing number is: " + findMissingNumber(arr) : "No missing number in the range 1 to 10.");
    }
}
```

```
[Running] cd "a:\Programs\HTML & CSS (from Sems)\4th Semester\CSW-2\05-03-2025 (Chap-13) ASSIGNMENT 2.1 & 2.2\ASSIGNMENT 2.2\" && javac Q10.java && java Q10
The smallest missing number is: 1

[Done] exited with code=0 in 1.057 seconds
```

Q11. Declare an array of integers: int[] arr = {1, 2, 10, 8, 7, 3, 4, 6, 5, 9};. Then, create a min-heap using the PriorityQueue class to store the elements from the array. Finally, dequeue the elements from the PriorityQueue using the appropriate methods and print them

Solution:


Q1.java
Q2.java
Q3.java
Q4.java
Q5.java
Q6.java
Q7.java
Q8.java
Q9.java
Q10.java
Q11.java
Q11.java > Q11 > main(String[])
1 import java.util.*;
2 public class Q11 {
Run | Debug
3 public static void main(String[] args) {
4 int[] arr = {7, 3, 9, 1, 5, 8, 2, 10, 6, 4};
5
6 // Create a min-heap (PriorityQueue)
7 PriorityQueue<Integer> minHeap = new PriorityQueue<>();
8
9 // Insert elements into the min-heap
10 for (int i = 0; i < arr.length; i++) {
11 minHeap.add(arr[i]);
12 }
13
14 // Dequeue elements and print them
15 System.out.println(x:"Elements in sorted order:");
16 while (!minHeap.isEmpty()) {
17 System.out.print(minHeap.poll() + " ");
18 }
19 }
20 }
PROBLEMS 2
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
[Running] cd "a:\Programs\HTML & CSS (from Sems)\4th Semester\CSW-2\05-03-2025 (Chap-13) ASSIGNMENT 2.1 & 2.2\ASSIGNMENT 2.2\"
Elements in sorted order:
1 2 3 4 5 6 7 8 9 10
[Done] exited with code=0 in 1.08 seconds