Rajalakshmi Engineering College

Name: Abiram G
Email: 241801005@rajalakshmi.edu.in
Roll no:
Phone: 9841900002
Branch: REC
Department: AI & DS - Section 5
Batch: 2028
Degree: B.E - AI & DS

Scan to verify results

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 10_Q4

Attempt : 1
Total Mark : 10
Marks Obtained : 9

Section 1 : COD

1. Problem Statement

In a ticket reservation system, you store the available seat numbers in a TreeSet. Users input their desired seat number, and the program checks whether the chosen seat is available.

Using a TreeSet ensures quick and efficient verification of seat availability, ensuring a smooth and organized ticket booking process.

Input Format

The first line of input contains a single integer n, representing the number of available seats.

The second line contains n space-separated integers, representing the available seat numbers.

The third line contains an integer m, representing the seat number that needs to be searched.

Output Format

The output displays "[m] is present!" if the given seat is available. Otherwise, it displays "[m] is not present!"

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 4
2 4 5 6
5
Output: 5 is present!

Answer

```java
// You are using Java
import java.util.Scanner;
import java.util.TreeSet;
class TicketReservationSystem {
  public static void main(String[] args) {
    Scanner scanner = new Scanner(System.in);

    int n = Integer.parseInt(scanner.nextLine());
    TreeSet<Integer> availableSeats = new TreeSet<>();

    String[] seatNumbers = scanner.nextLine().split(" ");
    for (String seat : seatNumbers) {
      availableSeats.add(Integer.parseInt(seat));
    }

    int m = Integer.parseInt(scanner.nextLine());

    if (availableSeats.contains(m)) {
      System.out.println(m + " is present!");
    } else {
      System.out.println(m + " is not present!");
    }
```

```
        scanner.close();
    }
}
```

Status : Partially correct   **Marks : 9/10**

Rajalakshmi Engineering College

Name: Abiram G
Email: 241801005@rajalakshmi.edu.in
Roll no:
Phone: 9841900002
Branch: REC
Department: AI & DS - Section 5
Batch: 2028
Degree: B.E - AI & DS

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 10_Q3

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : COD

1. Problem Statement

Priya is analyzing encrypted messages in a research project. She wants to analyze the frequency of each character in a given paragraph. The characters should be stored in a TreeMap so that the output is sorted in ascending order of characters automatically.

You are required to build a Java program that:

Uses a TreeMap<Character, Integer> to count how many times each character appears in the message.Ignores spaces and considers only alphabets (case-sensitive).Outputs the frequencies of characters in sorted order.

You must use a TreeMap in the class named MessageAnalyzer.

Input Format

The first line of input contains an integer n, the number of lines in the message.

The next n lines each contain a string (the encrypted message line).

Output Format

The first line of output prints: "Character Frequency:"

Then print each character and its frequency in the format: "<character>: <count>"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 2
Hello World
Java

Output: Character Frequency:
H: 1
J: 1
W: 1
a: 2
d: 1
e: 1
l: 3
o: 2
r: 1
v: 1

Answer

```java
// You are using Java
import java.util.Scanner;
import java.util.TreeMap;
class MessageAnalyzer {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        // Read the number of lines
        int n = Integer.parseInt(scanner.nextLine());
        TreeMap<Character, Integer> charFrequency = new TreeMap<>();
```

```java
        // Read each line and update character frequencies
        for (int i = 0; i < n; i++) {
            String line = scanner.nextLine();
            for (char ch : line.toCharArray()) {
                // Check if character is an alphabet
                if (Character.isLetter(ch)) {
                    charFrequency.put(ch, charFrequency.getOrDefault(ch, 0) + 1);
                }
            }
        }

        // Output the results
        System.out.println("Character Frequency:");
        for (char ch : charFrequency.keySet()) {
            System.out.println(ch + ": " + charFrequency.get(ch));
        }

        scanner.close();
    }
}
```

Status : Correct   *Marks : 10/10*

Rajalakshmi Engineering College

Name: Abiram G
Email: 241801005@rajalakshmi.edu.in
Roll no:
Phone: 9841900002
Branch: REC
Department: AI & DS - Section 5
Batch: 2028
Degree: B.E - AI & DS

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 10_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 6.5

Section 1 : COD

1. Problem Statement

John is organizing a fruit festival, and the quantities of various fruits are stored in a HashMap where fruit names are keys and quantities are values.

Help him develop a program to find the total quantity of fruits for the festival by summing up the values in the HashMap.

Input Format

The input consists of fruit quantities in the format 'fruitName:quantity', where fruitName is the name of the fruit(a string), and quantity is a double value representing the quantity.

The input is terminated by entering "done".

Output Format

The output prints a double value, representing the sum of values in the HashMap, rounded off to two decimal places.

If the value is not numeric, print "Invalid input".

If any special characters other than ':' are entered, print "Invalid format".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Banana:15.2
Orange:56.3
Mango:47.3
done

Output: 118.80

Answer

```java
// You are using Java
import java.util.HashMap;
import java.util.Scanner;

class FruitFestival {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        HashMap<String, Double> fruitQuantities = new HashMap<>();
        String input;

        while (true) {
            input = scanner.nextLine();
            if (input.equals("done")) {
                break;
            }

            // Validate format
            if (!input.contains(":") || input.indexOf(":") != input.lastIndexOf(":")) {
                System.out.println("Invalid format");
                continue;
            }
```

```java
        String[] parts = input.split(":");
        String fruitName = parts[0].trim();
        String quantityStr = parts[1].trim();

        // Validate quantity
        try {
            double quantity = Double.parseDouble(quantityStr);
            if (quantity < 1.0 || quantity > 100.0) {
                System.out.println("Invalid input");
                continue;
            }
            fruitQuantities.put(fruitName, quantity);
        } catch (NumberFormatException e) {
            System.out.println("Invalid input");
        }
    }

    // Calculate total quantity
    double totalQuantity = 0.0;
    for (double quantity : fruitQuantities.values()) {
        totalQuantity += quantity;
    }

    // Print total quantity rounded to two decimal places
    System.out.printf("%.2f\n", totalQuantity);
    scanner.close();
  }
}
```

Status : Partially correct   *Marks : 6.5/10*

Rajalakshmi Engineering College

Name: Abiram G
Email: 241801005@rajalakshmi.edu.in
Roll no:
Phone: 9841900002
Branch: REC
Department: AI & DS - Section 5
Batch: 2028
Degree: B.E - AI & DS

2024_28_III_OOPS Using Java Lab

2028_REC_OOPS using Java_Week 10_Q1

Attempt : 1
Total Mark : 10
Marks Obtained : 10

Section 1 : COD

1. Problem Statement

A city traffic management system needs to track vehicles entering a toll booth. Each vehicle is uniquely identified by its registration number. The system should allow adding vehicles to a record, ensuring that no duplicate registration numbers exist. The vehicles should be stored in a HashSet, which does not guarantee any specific order.

Your task is to implement a program using a HashSet that allows adding vehicle details and displaying the records.

Input Format

The first line of input contains an integer N - the number of vehicles.

The next N lines contain details of each vehicle in the format: "RegNumber

OwnerName VehicleType"

1. RegNumber (String) - A unique registration number (Alphanumeric).
2. OwnerName (String) - The name of the vehicle owner.
3. VehicleType (String, Car, Bike, or Truck) - The type of vehicle.

If a vehicle with the same registration number is already present, ignore the duplicate entry.

Output Format

The output prints the unique vehicle records in any order (since HashSet does not maintain order).

Output format: "RegNumber OwnerName VehicleType"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
KA01AB1234 John Car
MH02CD5678 Alice Bike
DL03EF9012 Bob Truck
TN04GH3456 Mike Car
KA01AB1234 John Car

Output: TN04GH3456 Mike Car
KA01AB1234 John Car
MH02CD5678 Alice Bike
DL03EF9012 Bob Truck

Answer

```java
import java.util.HashSet;
import java.util.Scanner;

class Vehicle {
    String regNumber;
    String ownerName;
    String vehicleType;

    public Vehicle(String regNumber, String ownerName, String vehicleType) {
```

```java
        this.regNumber = regNumber;
        this.ownerName = ownerName;
        this.vehicleType = vehicleType;
    }

    @Override
    public String toString() {
        return regNumber + " " + ownerName + " " + vehicleType;
    }

    @Override
    public int hashCode() {
        return regNumber.hashCode();
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (!(obj instanceof Vehicle)) return false;
        Vehicle other = (Vehicle) obj;
        return this.regNumber.equals(other.regNumber);
    }
}

class TrafficManagementSystem {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        HashSet<Vehicle> vehicles = new HashSet<>();

        // Read number of vehicles
        int n = Integer.parseInt(scanner.nextLine());

        // Read vehicle details
        for (int i = 0; i < n; i++) {
            String input = scanner.nextLine();
            String[] details = input.split(" ");
            String regNumber = details[0];
            String ownerName = details[1];
            String vehicleType = details[2];

            Vehicle vehicle = new Vehicle(regNumber, ownerName, vehicleType);
            vehicles.add(vehicle);
```

```java
        }

        // Display unique vehicle records
        for (Vehicle vehicle : vehicles) {
            System.out.println(vehicle);
        }

        scanner.close();
    }
}
```

Status : Correct   **Marks : 10/10**

Rajalakshmi Engineering College

Name: Abiram G
Email: 241801005@rajalakshmi.edu.in
Roll no:
Phone: 9841900002
Branch: REC
Department: AI & DS - Section 5
Batch: 2028
Degree: B.E - AI & DS

2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 10_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 10

Section 1 : MCQ

1. How does HashSet check for duplicate elements?

Answer

Using equals() and hashCode()

Status : Correct   *Marks : 1/1*

2. Which of the following is true about HashMap?

Answer

It does not allow duplicate values

Status : Wrong   *Marks : 0/1*

3. What will be the output of the following code?

```java
import java.util.*;
class Main {
    public static void main(String[] args) {
        HashMap<String, Integer> map = new HashMap<>();
        map.put("A", 1);
        map.put("B", 2);
        map.put("C", 3);
        System.out.println(map.containsKey("B"));
    }
}
```

Answer

true

Status : Correct   *Marks : 1/1*

4. What is the time complexity of retrieving an element from a HashSet?

Answer

O(1)

Status : Correct   *Marks : 1/1*

5. Which of the following is true about TreeMap?

Answer

It maintains natural ordering

Status : Correct   *Marks : 1/1*

6. What will be the output of the following code?

```java
import java.util.*;
class Main {
    public static void main(String[] args) {
        HashMap<String, Integer> map = new HashMap<>();
```

```
    map.put("X", 10);
    map.put("Y", 20);
    map.put("Z", 30);
    map.remove("Y");
    System.out.println(map);
  }
}
```

Answer

{X=10, Z=30}

Status : Correct   *Marks : 1/1*

7. What will be the output of the following code?

```
import java.util.*;
class Main {
  public static void main(String[] args) {
    HashMap<String, String> map = new HashMap<>();
    map.put("A", "Apple");
    map.put("B", "Banana");
    map.put("C", "Cherry");
    map.replace("B", "Blueberry");
    System.out.println(map);
  }
}
```

Answer

{A=Apple, B=Blueberry, C=Cherry}

Status : Correct   *Marks : 1/1*

8. What happens when you add duplicate elements to a HashSet?

Answer

The duplicate is ignored

Status : Correct   *Marks : 1/1*

9. Which of the following allows null keys in Java?

Answer

HashMap

Status : Correct   *Marks : 1/1*

10. Which method removes all elements from a Set?

Answer

clear()

Status : Correct   *Marks : 1/1*

11. What will happen if you add elements in descending order in a TreeSet?

Answer

An exception occurs

Status : Wrong   *Marks : 0/1*

12. What happens if two keys have the same hash code in a HashMap?

Answer

The existing value gets replaced

Status : Wrong   *Marks : 0/1*

13. Which method retrieves the lowest key in a TreeMap?

Answer

lowerKey()

Status : Wrong   *Marks : 0/1*

14. Which statement is true about HashSet and TreeSet?

Answer

Both maintain order

Status : Wrong   *Marks : 0/1*

15. What will happen if you add a null element to a TreeSet?

Answer

An exception occurs

Status : Correct   *Marks : 1/1*