

Rajalakshmi Engineering College

Name: Tarika A G
Email: 241801290@rajalakshmi.edu.in
Roll no: 241801290
Phone: 8807807801
Branch: REC
Department: AI & DS - Section 3
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_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

```
import java.util.*;  
class Vehicle {  
    String regNumber;  
    String ownerName;  
    String vehicleType;  
    Vehicle(String regNumber, String ownerName, String vehicleType) {  
        this.regNumber = regNumber;  
        this.ownerName = ownerName;  
        this.vehicleType = vehicleType;
```

```
    }
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null || getClass() != obj.getClass())
            return false;
        Vehicle v = (Vehicle) obj;
        return regNumber.equals(v.regNumber);
    }
    public int hashCode() {
        return regNumber.hashCode();
    }
    public String toString() {
        return regNumber + " " + ownerName + " " + vehicleType;
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        HashSet<Vehicle> vehicles = new HashSet<>();
        for (int i = 0; i < n; i++) {
            String reg = sc.next();
            String owner = sc.next();
            String type = sc.next();
            vehicles.add(new Vehicle(reg, owner, type));
        }
        for (Vehicle v : vehicles) {
            System.out.println(v);
        }
    }
}
```

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Tarika A G
Email: 241801290@rajalakshmi.edu.in
Roll no: 241801290
Phone: 8807807801
Branch: REC
Department: AI & DS - Section 3
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_Q2

Attempt : 1
Total Mark : 10
Marks Obtained : 10

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

```
import java.util.*;
import java.text.DecimalFormat;
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        HashMap<String, Double> fruits = new HashMap<>();
        boolean invalidFormat = false;
        boolean invalidInput = false;
        double total = 0.0;
        while (sc.hasNext()) {
            String input = sc.next();
            if (input.equalsIgnoreCase("done")) {
                break;
            }
            if (!input.contains(":")) {
                invalidFormat = true;
                break;
            }
            String[] parts = input.split(":");
            if (parts.length != 2) {
                invalidFormat = true;
                break;
            }
            try {
                Double value = Double.parseDouble(parts[1]);
                fruits.put(parts[0], value);
                total += value;
            } catch (NumberFormatException e) {
                invalidInput = true;
            }
        }
        if (invalidFormat) {
            System.out.println("Invalid format");
        } else if (invalidInput) {
            System.out.println("Invalid input");
        } else {
            DecimalFormat df = new DecimalFormat("0.00");
            System.out.println(df.format(total));
        }
    }
}
```

```
        }
        String fruit = parts[0];
        String quantityStr = parts[1];
        if (!fruit.matches("[A-Za-z]+")) {
            invalidFormat = true;
            break;
        }
        try {
            double quantity = Double.parseDouble(quantityStr);
            fruits.put(fruit, quantity);
        } catch (NumberFormatException e) {
            invalidInput = true;
            break;
        }
    }
    if (invalidFormat) {
        System.out.print("Invalid format");
    } else if (invalidInput) {
        System.out.print("Invalid input");
    } else {
        for (double value : fruits.values()) {
            total += value;
        }
        DecimalFormat df = new DecimalFormat("0.00");
        System.out.print(df.format(total));
    }
}
```

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Tarika A G
Email: 241801290@rajalakshmi.edu.in
Roll no: 241801290
Phone: 8807807801
Branch: REC
Department: AI & DS - Section 3
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_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

```
import java.util.*;  
class MessageAnalyzer {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
        int n = sc.nextInt();  
        sc.nextLine();  
        TreeMap<Character, Integer> freqMap = new TreeMap<>();  
        for (int i = 0; i < n; i++) {  
            String line = sc.nextLine();  
            for (char ch : line.toCharArray()) {  
                if (Character.isAlphabetic(ch)) {
```

```
        freqMap.put(ch, freqMap.getOrDefault(ch, 0) + 1);
    }
}
System.out.print("Character Frequency: ");
for (Map.Entry<Character, Integer> entry : freqMap.entrySet()) {
    System.out.print(entry.getKey() + ":" + entry.getValue() + " ");
}
}
```

Status : Correct

Marks : 10/10

Rajalakshmi Engineering College

Name: Tarika A G
Email: 241801290@rajalakshmi.edu.in
Roll no: 241801290
Phone: 8807807801
Branch: REC
Department: AI & DS - Section 3
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 : 10

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

```
import java.util.*;
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        TreeSet<Integer> seats = new TreeSet<>();
        for (int i = 0; i < n; i++) {
            seats.add(sc.nextInt());
        }
        int m = sc.nextInt();
        if (seats.contains(m)) {
            System.out.print(m + " is present!");
        } else {
            System.out.print(m + " is not present!");
        }
    }
}
```

Status : Correct

Marks : 10/10