# Rajalakshmi Engineering College

Name: Suria Prakash
Email: 241501225@rajalakshmi.edu.in
Roll no: 241501225
Phone: 8056422252
Branch: REC
Department: AI & ML - Section 4
Batch: 2028
Degree: B.E - AI & ML

Scan to verify results

## 2024_28_III_OOPS Using Java Lab

### REC_2028_OOPS using Java_Week 10_MCQ

Attempt : 1
Total Mark : 15
Marks Obtained : 15

### Section 1 : MCQ

1. Which of the following is true about TreeMap?

***Answer***

It maintains natural ordering

***Status :*** Correct ***Marks : 1/1***

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

***Answer***

HashMap

***Status :*** Correct ***Marks : 1/1***

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

***Answer***

O(1)

***Status :*** Correct ***Marks : 1/1***

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

***Answer***

They are sorted in ascending order

***Status :*** Correct ***Marks : 1/1***

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

***Answer***

An exception occurs

***Status :*** Correct ***Marks : 1/1***

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

```
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***

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 if two keys have the same hash code in a HashMap?

***Answer***

A linked list is used to store values with the same hash

***Status :*** Correct ***Marks : 1/1***

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

***Answer***

firstKey()

***Status :*** Correct ***Marks : 1/1***

10. How does HashSet check for duplicate elements?

***Answer***

Using equals() and hashCode()

***Status :*** Correct **Marks : 1/1**

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

***Answer***

The duplicate is ignored

***Status :*** Correct **Marks : 1/1**

12. Which of the following is true about HashMap?

***Answer***

It is not synchronized

***Status :*** Correct **Marks : 1/1**

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

```
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**

14. Which statement is true about HashSet and TreeSet?

***Answer***

TreeSet provides sorted elements

***Status :*** Correct ***Marks : 1/1***

15. Which method removes all elements from a Set?

***Answer***

clear()

***Status :*** Correct ***Marks : 1/1***

# Rajalakshmi Engineering College

Name: Suria Prakash
Email: 241501225@rajalakshmi.edu.in
Roll no: 241501225
Phone: 8056422252
Branch: REC
Department: AI & ML - Section 4
Batch: 2028
Degree: B.E - AI & ML

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***

```
// You are using Java
import java.util.HashSet;
import java.util.Scanner;
import java.util.Set;
import java.util.Objects;

class Vehicle {
    String regNumber;
    String ownerName;
```

```
    String vehicleType;

    Vehicle(String regNumber, String ownerName, String vehicleType) {
        this.regNumber = regNumber;
        this.ownerName = ownerName;
        this.vehicleType = vehicleType;
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (!(o instanceof Vehicle)) return false;
        Vehicle v = (Vehicle) o;
        // Uniqueness is determined by registration number
        return regNumber.equals(v.regNumber);
    }

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

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

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = Integer.parseInt(sc.nextLine().trim());

        Set<Vehicle> vehicles = new HashSet<>();

        for (int i = 0; i < N; i++) {
            String line = sc.nextLine();
            // Expect: "RegNumber OwnerName VehicleType"
            // Split into three parts; OwnerName may contain spaces? As per
constraints, assume single token.
            String[] parts = line.trim().split("\\s+");
            if (parts.length < 3) continue; // guard against malformed input
```

```
            String regNumber = parts[0];
            String ownerName = parts[1];
            String vehicleType = parts[2];

            Vehicle v = new Vehicle(regNumber, ownerName, vehicleType);
            vehicles.add(v); // duplicates ignored due to equals/hashCode
        }

        // Print unique records in any order
        for (Vehicle v : vehicles) {
            System.out.println(v.toString());
        }

        sc.close();
    }
}
```

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

# Rajalakshmi Engineering College

Name: Suria Prakash
Email: 241501225@rajalakshmi.edu.in
Roll no: 241501225
Phone: 8056422252
Branch: REC
Department: AI & ML - Section 4
Batch: 2028
Degree: B.E - AI & ML

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***

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

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Map<String, Double> map = new HashMap<>();
        boolean invalidFormat = false;
        boolean invalidInput = false;

        while (sc.hasNextLine()) {
            String line = sc.nextLine().trim();
            if (line.equalsIgnoreCase("done")) {
                break;
            }

            // Check for invalid characters: only allowed ':' as separator
            // If the line contains characters other than letters, digits, colon, and dot,
be strict:
            // We specifically detect if it contains '-' or other punctuation besides ':'.
```

```
if (line.contains("-") || line.contains(",")) {
    System.out.println("Invalid format");
    sc.close();
    return;
}

// Split by colon
int colonIndex = line.indexOf(':');
if (colonIndex == -1 || line.indexOf(':', colonIndex + 1) != -1) {
    // either no colon or multiple colons
    System.out.println("Invalid format");
    sc.close();
    return;
}

String fruitName = line.substring(0, colonIndex).trim();
String quantityStr = line.substring(colonIndex + 1).trim();

// Validate fruitName length (1 to 20)
if (fruitName.length() < 1 || fruitName.length() > 20) {
    System.out.println("Invalid format");
    sc.close();
    return;
}

// Validate quantity is numeric
double quantity;
try {
    quantity = Double.parseDouble(quantityStr);
} catch (NumberFormatException e) {
    System.out.println("Invalid input");
    sc.close();
    return;
}

// Check range of quantity
if (quantity < 1.0 || quantity > 100.0) {
    System.out.println("Invalid input");
    sc.close();
    return;
}
```

```
        // Store in map (if duplicates appear, update to last value or ignore?
Problem statement sums values, duplicates unlikely.
            map.put(fruitName, quantity);
        }

        // Sum all values
        double sum = 0.0;
        for (double v : map.values()) {
            sum += v;
        }

        // Print with two decimals
        System.out.printf("%.2f%n", sum);
        sc.close();
    }
}
```

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

# Rajalakshmi Engineering College

Name: Suria Prakash
Email: 241501225@rajalakshmi.edu.in
Roll no: 241501225
Phone: 8056422252
Branch: REC
Department: AI & ML - Section 4
Batch: 2028
Degree: B.E - AI & ML

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***

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

class MessageAnalyzer {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = 0;
        if (sc.hasNextLine()) {
            String line = sc.nextLine().trim();
```

```
        try {
            n = Integer.parseInt(line);
        } catch (NumberFormatException e) {
            // If the first line isn't a valid number, treat as zero lines
            n = 0;
        }
    }

    TreeMap<Character, Integer> freq = new TreeMap<>();

    for (int i = 0; i < n && sc.hasNextLine(); i++) {
        String msg = sc.nextLine();
        for (int j = 0; j < msg.length(); j++) {
            char ch = msg.charAt(j);
            if (Character.isLetter(ch)) {
                freq.put(ch, freq.getOrDefault(ch, 0) + 1);
            }
        }
    }

    System.out.println("Character Frequency:");
    for (Map.Entry<Character, Integer> e : freq.entrySet()) {
        System.out.println(e.getKey() + ": " + e.getValue());
    }

    sc.close();
  }
}
```

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