

Rajalakshmi Engineering College

Name: Jesvanth Sabarish V K
Email: 240701214@rajalakshmi.edu.in
Roll no: 240701214
Phone: 9080128264
Branch: REC
Department: CSE - Section 6
Batch: 2028
Degree: B.E - CSE

Scan to verify results



2024_28_III_OOPS Using Java Lab

REC_2028_OOPS using Java_Week 10_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 10

Section 1 : COD

1. Problem Statement

Bob wants to develop a score-tracking application for a gaming tournament. Each player's score is stored in a HashMap with the player's name as the key and the score as the value.

Write a program to assist Bob that takes user input to enter player scores, calculates the maximum score from the HashMap, and prints the player with the highest score.

Input Format

The input consists of strings representing player details in the format "playerName:score".

The input is terminated by entering "done".

Output Format

The output displays a string, representing the player's name who scored the maximum.

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

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

Refer to the sample output for formatting specifications.

Sample Test Case

Input: Alice:15

Bob:56

done

Output: Bob

Answer

-

Status : Skipped

Marks : 0/10

2. Problem Statement

David is managing an employee database where each employee has a unique ID, name, and department. He wants to ensure that duplicate employee IDs are not added to the system. Implement a Java program that allows adding employees to the system, displaying all employees, and checking if an employee exists based on the given ID.

Implement a class EmployeeDatabase that contains a HashSet to store employee records. The Employee class should be a user-defined object containing employee details. The main class should handle user operations and interact with the EmployeeDatabase class.

Input Format

The first line contains an integer n representing the number of employees to be added.

The next n lines follow, each containing:

1. An integer employee_id
2. A string name
3. A string department

The next line contains an integer m representing the number of queries.

The next m lines follow, each containing an employee ID to check for existence.

Output Format

The output prints a list of all employees added in the format:

"ID: <employee_id>, Name: <name>, Department: <department>"

For each query, output "Employee exists" if the ID is found, otherwise "Employee not found".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3

101 John IT

102 Alice HR

103 Bob Finance

2

101

104

Output: ID: 101, Name: John, Department: IT

ID: 102, Name: Alice, Department: HR

ID: 103, Name: Bob, Department: Finance

Employee exists

Employee not found

Answer

```
import java.util.*;
```

```
// You are using Java
```

```

class Employee {
    //Type your code here
}

class EmployeeDatabase {
    HashSet
    //Type your code here
}

class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        EmployeeDatabase db = new EmployeeDatabase();
        int n = sc.nextInt();
        for (int i = 0; i < n; i++) {
            int id = sc.nextInt();
            String name = sc.next();
            String department = sc.next();
            db.addEmployee(id, name, department);
        }
        db.displayEmployees();
        int m = sc.nextInt();
        for (int i = 0; i < m; i++) {
            int id = sc.nextInt();
            if (db.checkEmployee(id))
                System.out.println("Employee exists");
            else
                System.out.println("Employee not found");
        }
        sc.close();
    }
}

```

Status : Wrong

Marks : 0/10

3. Problem Statement

A college professor wants to keep track of students who attend classes. Each student has a unique roll number and their attendance count increases every time they attend a class. The system should allow adding a student, marking their attendance, and displaying all students with their

total attendance.

Your task is to implement a Java program using TreeSet to maintain students in sorted order of roll numbers and track their attendance count.

Operations:

A roll_no name Add a student with roll number and name (if not already added). M roll_no Mark attendance for the student with the given roll number (increase their count by 1). D Display all students in ascending order of roll number along with their attendance count.

Input Format

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

The next N lines contain one of the following commands:

A roll_no name

M roll_no

D

- A (Add) Adds a new student with a unique roll number and name.
- M (Mark) Increases attendance count for the given roll number.
- D (Display) Prints all students in ascending order of roll number.

Output Format

For D, output prints each student's roll number, name, and attendance count in ascending order of roll number.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5

A 101 Alice

A 102 Bob

M 101

M 101

D

Output: 101 Alice 2
102 Bob 0

Answer

// You are using Java
import java.util.*;

```
class Student implements Comparable<Student> {  
    int roll;  
    String name;  
    int attendance;  
  
    Student(int roll, String name) {  
        this.roll = roll;  
        this.name = name;  
        this.attendance = 0;  
    }  
  
    @Override  
    public int compareTo(Student other) {  
        return Integer.compare(this.roll, other.roll);  
    }  
}
```

```
public class Main {  
    public static void main(String[] args) {  
        Scanner sc = new Scanner(System.in);  
  
        int N = Integer.parseInt(sc.nextLine());  
  
        TreeSet<Student> set = new TreeSet<>();  
  
        for (int i = 0; i < N; i++) {  
            String line = sc.nextLine();  
            String[] parts = line.split(" ");  
  
            char cmd = parts[0].charAt(0);  
  
            if (cmd == 'A') {  
                int roll = Integer.parseInt(parts[1]);  
                String name = parts[2];
```

```

        // Check if student already exists
        Student temp = new Student(roll, name);
        if (!set.contains(temp)) {
            set.add(temp);
        }

    } else if (cmd == 'M') {
        int roll = Integer.parseInt(parts[1]);

        for (Student s : set) {
            if (s.roll == roll) {
                s.attendance++;
                break;
            }
        }

    } else if (cmd == 'D') {
        for (Student s : set) {
            System.out.println(s.roll + " " + s.name + " " + s.attendance);
        }
    }
}
}
}
}
}

```

Status : Correct

Marks : 10/10

4. Problem Statement

Tony is an e-learning platform administrator, he oversees the user ratings for various online courses offered in the platform.

To enhance user experience, you should assist him in utilizing a HashMap to store course ratings given by learners. Regularly, he analyzes this data to identify the highest and lowest-rated courses, enabling targeted improvements and ensuring the quality of the educational content. This process assists in maintaining a competitive and engaging online learning environment for the users.

Input Format

The input consists of a string representing the course name followed by a double value representing the course's rating, in separate lines.

The input is terminated by entering "done".

Output Format

The first line of output prints the string "Highest Rated Course: " followed by the highest-rated course.

The second line prints the string "Lowest Rated Course: " followed by the lowest-rated courses.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: DSA

4.0

OOPS

4.2

C

3.2

done

Output: Highest Rated Course: OOPS

Lowest Rated Course: C

Answer

-

Status : Skipped

Marks : 0/10