Rajalakshmi Engineering College

Name: Adithya B
Email: 240701018@rajalakshmi.edu.in
Roll no: 240701018
Phone: 9444117405
Branch: REC
Department: CSE - Section 10
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 : 40

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

```
import java.util.*;

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

class ScoreTracker {
    HashMap<String, Integer> scoreMap = new HashMap<>();

    public boolean processInput(String input) {

        if (!input.contains(":") || input.indexOf(':') != input.lastIndexOf(':')) {
            System.out.println("Invalid format");
            return false;
        }

        String[] parts = input.split(":");
        if (parts.length != 2) {
            System.out.println("Invalid format");
            return false;
        }

        String name = parts[0];
```

```
        String scoreStr = parts[1];

        if (!name.matches("[A-Za-z]+")) {
            System.out.println("Invalid format");
            return false;
        }


        int score;
        try {
            score = Integer.parseInt(scoreStr);
        } catch (Exception e) {
            System.out.println("Invalid input");
            return false;
        }


        scoreMap.put(name, score);
        return true;
    }

    public String findTopPlayer() {
        String topPlayer = "";
        int maxScore = Integer.MIN_VALUE;

        for (String player : scoreMap.keySet()) {
            int s = scoreMap.get(player);
            if (s > maxScore) {
                maxScore = s;
                topPlayer = player;
            }
        }
        return topPlayer;
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        ScoreTracker tracker = new ScoreTracker();
        boolean validInput = true;

        while (true) {
```

```
            String input = scanner.nextLine();

            if (input.toLowerCase().equals("done")) {
                break;
            }

            if (!tracker.processInput(input)) {
                validInput = false;
                break;
            }
        }

        if (validInput && !tracker.scoreMap.isEmpty()) {
            System.out.println(tracker.findTopPlayer());
        }

        scanner.close();
    }
}
```

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

2. Problem Statement

A linguist named Meera is classifying a list of words based on their first character. She wants to store words grouped by their starting letter using a TreeMap so that the groups appear in sorted order of characters (i.e., 'a' to 'z'). For each letter, all words starting with that letter should be stored in the order they appear.

Implement the logic inside a class named WordClassifier using the TreeMap<Character, List<String>> collection.

Input Format

The first line of the input contains an integer n, representing the number of words.

The next n lines each contain a word.

Output Format

The first line of the output prints: "Grouped Words by Starting Letter:"

The next lines print each character key and its list of words in the format:

"letter: word1 word2 word3...

..."

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
dog
deer
cat
cow
camel
Output: Grouped Words by Starting Letter:
c: cat cow camel
d: dog deer

Answer

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

import java.util.*;

```
class WordClassifier {

   TreeMap<Character, List<String>> map = new TreeMap<>();

   public void classifyWords(List<String> words) {

      for (String w : words) {
         char ch = w.charAt(0);

         if (!map.containsKey(ch)) {
            map.put(ch, new ArrayList<>());
         }
```

```
            map.get(ch).add(w);
        }

        System.out.println("Grouped Words by Starting Letter:");

        for (Map.Entry<Character, List<String>> entry : map.entrySet()) {
            System.out.print(entry.getKey() + ": ");
            for (String s : entry.getValue()) {
                System.out.print(s + " ");
            }
            System.out.print(" ");
        }
    }
}
public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = Integer.parseInt(sc.nextLine());

        List<String> words = new ArrayList<>();
        for (int i = 0; i < n; i++) {
            words.add(sc.nextLine());
        }

        WordClassifier classifier = new WordClassifier();
        classifier.classifyWords(words);
    }
}
```

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

3. Problem Statement

The city library maintains a record of books available for lending. Each book is uniquely identified by its ISBN number, along with its title and author. The librarian wants to efficiently store and manage these records, ensuring books can be listed in the order they were added.

Your task is to implement a Library Management System using HashSet where:

The librarian adds books with ISBN, title, and author.The librarian can remove books by providing an ISBN.Finally, the librarian displays the available books in the order they were added.

Implement a class Library that will handle these operations. The main function should manage user input and interact with the Library class accordingly.

Input Format

The first line contains an integer n – the number of books to be added.

The next n lines contain three values: ISBN (integer), Title (string without spaces), and Author (string without spaces).

1. An integer employee_id
2. A string title
3. A string author name

The next line contains an integer m – the number of books to be removed.

The next m lines follow, each contains an ISBN number to remove.

Output Format

The output prints a list of books available in the library after performing all operations in the format:

"ISBN: <isbn>, Title: <title>, Author: <author>"

If no books remain, print: "No books available"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 3
1234 JavaCompleteGuide JohnDoe
5678 PythonBasics JaneDoe
9012 DataStructures AliceSmith
1

5679

Output: ISBN: 1234, Title: JavaCompleteGuide, Author: JohnDoe
ISBN: 9012, Title: DataStructures, Author: AliceSmith
ISBN: 5678, Title: PythonBasics, Author: JaneDoe

Answer

```
import java.util.*;

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

class Book {
    int isbn;
    String title;
    String author;

    public Book(int isbn, String title, String author) {
        this.isbn = isbn;
        this.title = title;
        this.author = author;
    }

    @Override
    public boolean equals(Object obj) {
        if (this == obj) return true;
        if (!(obj instanceof Book)) return false;
        Book b = (Book) obj;
        return this.isbn == b.isbn;
    }

    @Override
    public int hashCode() {
        return Objects.hash(isbn);
    }
}

class Library {
    HashSet<Book> books = new HashSet<>();

    public void addBook(int isbn, String title, String author) {
        books.add(new Book(isbn, title, author));
    }
```

```
    public void removeBook(int isbn) {
        Book toRemove = null;
        for (Book b : books) {
            if (b.isbn == isbn) {
                toRemove = b;
                break;
            }
        }
        if (toRemove != null) books.remove(toRemove);
    }

    public void displayBooks() {
        if (books.isEmpty()) {
            System.out.println("No books available");
            return;
        }

        for (Book b : books) {
            System.out.print("ISBN: " + b.isbn + ", Title: " + b.title + ", Author: " +
b.author + " ");
        }
    }
}


class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        Library library = new Library();
        int n = sc.nextInt();
        for (int i = 0; i < n; i++) {
            int isbn = sc.nextInt();
            String title = sc.next();
            String author = sc.next();
            library.addBook(isbn, title, author);
        }
        int m = sc.nextInt();
        for (int i = 0; i < m; i++) {
            int isbn = sc.nextInt();
            library.removeBook(isbn);
        }
        library.displayBooks();
```

```
        sc.close();
    }
}
```

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

```
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

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

class CourseAnalyzer {

    public Map<String, String>
identifyHighestAndLowestRatedCourses(Map<String, Double> courseRatings) {

        String highestCourse = "";
        String lowestCourse = "";
        double highest = Double.MIN_VALUE;
        double lowest = Double.MAX_VALUE;

        for (Map.Entry<String, Double> entry : courseRatings.entrySet()) {
            String course = entry.getKey();
            double rating = entry.getValue();

            if (rating > highest) {
                highest = rating;
                highestCourse = course;
            }

            if (rating < lowest) {
                lowest = rating;
                lowestCourse = course;
            }
        }

        Map<String, String> result = new HashMap<>();
```

```
        result.put("highest", highestCourse);
        result.put("lowest", lowestCourse);

        return result;
    }



}
public class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        Map<String, Double> courseRatings = new HashMap<>();

        while (true) {
            String courseName = scanner.nextLine();
            if (courseName.equalsIgnoreCase("done")) {
                break;
            }
            double rating = Double.parseDouble(scanner.nextLine().trim());
            courseRatings.put(courseName, rating);
        }

        CourseAnalyzer analyzer = new CourseAnalyzer();
        Map<String, String> result =
analyzer.identifyHighestAndLowestRatedCourses(courseRatings);

        System.out.printf("Highest Rated Course: %s\n", result.get("highest"));
        System.out.printf("Lowest Rated Course: %s", result.get("lowest"));

        scanner.close();
    }
}
```

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