

Rajalakshmi Engineering College

Name: DARSHAN M
Email: 241501040@rajalakshmi.edu.in
Roll no: 241501040
Phone: 9360697087
Branch: REC
Department: AI & ML - Section 1
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_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : COD

1. Problem Statement

Aryan is developing a voting system for a college election. Each vote is recorded as an entry in an array, where every student's vote is represented by a candidate's ID. Since it's a majority-rule election, the winner is the candidate who receives more than $n/2$ votes, where n is the total number of votes cast.

To quickly determine the winner, Aryan decides to use a HashMap to count the occurrences of each vote and identify the candidate who has received more than half of the total votes.

Example

Input

7

2 2 1 2 2 2 3

Output

2

Explanation

The votes are: 2, 2, 1, 2, 2, 3, 2

Count of each candidate:

2 appears 5 times 1 appears once 3 appears once

The majority element is the one that appears more than $N/2$ times. Since $7/2 = 3.5$, a number must appear at least 4 times to be the majority.

The number 2 appears 5 times, which is greater than 3.5, so the output is 2.

Input Format

The first line contains an integer N representing the number of votes cast.

The second line contains N space-separated integers representing the votes, where each integer corresponds to a candidate.

Output Format

The output prints an integer representing the majority element (the candidate who received more than $N/2$ votes).

If no such candidate exists, print -1.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 7

2 2 1 2 2 2 3

Output: 2

Answer

```
import java.util.HashMap;
```

```

import java.util.Scanner;

class MajorityElementFinder {
    public static int findMajorityElement(int[] arr) {
        int n = arr.length;
        HashMap<Integer, Integer> map = new HashMap<>();
        for (int num : arr) map.put(num, map.getOrDefault(num, 0) + 1);
        for (int key : map.keySet()) {
            if (map.get(key) > n / 2) return key;
        }
        return -1;
    }
}

class Main {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        int N = scanner.nextInt();
        int[] arr = new int[N];

        for (int i = 0; i < N; i++) {
            arr[i] = scanner.nextInt();
        }

        int result = MajorityElementFinder.findMajorityElement(arr);
        System.out.println(result);

        scanner.close();
    }
}

```

Status : Correct

Marks : 10/10

2. 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;
```

```
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()) {
        double rating = entry.getValue();
        if (rating > highest) {
            highest = rating;
            highestCourse = entry.getKey();
        }
        if (rating < lowest) {
            lowest = rating;
            lowestCourse = entry.getKey();
        }
    }

    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

3. 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.*;
```

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

```
class WordClassifier {
```

```
    public void classifyWords(List<String> words) {
```

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

```
        for (String word : words) {
```

```
            char firstChar = word.charAt(0);
```

```
            map.putIfAbsent(firstChar, new ArrayList<>());
```

```
            map.get(firstChar).add(word);
```

```
        }
```

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

```
        for (Map.Entry<Character, List<String>> entry : map.entrySet()) {
```

```
            System.out.print(entry.getKey() + ": ");
```

```
            for (String w : entry.getValue()) {
```

```
                System.out.print(w + " ");
```

```
            }
```

```
            System.out.println();
```

```
        }
```

```
    }
```

```
}
```

```
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

4. 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.*;
```

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

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

```

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

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

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

    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.println("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