Rajalakshmi Engineering College

Name: Mahalaxmi VPK
Email: 241901052@rajalakshmi.edu.in
Roll no: 241901052
Phone: 7904361227
Branch: REC
Department: CSE (CS) - Section 1
Batch: 2028
Degree: B.E - CSE (CS)

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 : 33.5

Section 1 : COD

1. 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 rollNo;
    String name;
    int attendance;
```

```
    public Student(int rollNo, String name) {
        this.rollNo = rollNo;
        this.name = name;
        this.attendance = 0;
    }
    @Override
    public int compareTo(Student other) {
        return Integer.compare(this.rollNo, other.rollNo);
    }
    @Override
    public boolean equals(Object o) {
        if (o instanceof Student) {
            return this.rollNo == ((Student) o).rollNo;
        }
        return false;
    }
    @Override
    public int hashCode() {
        return Objects.hash(rollNo);
    }
}
class StudentAttendance {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int N = Integer.parseInt(sc.nextLine());
        TreeSet<Student> students = new TreeSet<>();
        for (int i = 0; i < N; i++) {
            String[] parts = sc.nextLine().split(" ");
            if (parts[0].equals("A")) {
                int roll = Integer.parseInt(parts[1]);
                String name = parts[2];
                students.add(new Student(roll, name));
            } else if (parts[0].equals("M")) {
                int roll = Integer.parseInt(parts[1]);
                for (Student s : students) {
                    if (s.rollNo == roll) {
                s.attendance++;
                        break;
                    }
                }
            } else if (parts[0].equals("D")) {
```

```
            for (Student s : students) {
                System.out.println(s.rollNo + " " + s.name + " " + s.attendance);
            }
        }
    }
  }
}
```

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

2. Problem Statement

Arjun is working on a program that checks if one set of numbers is a subset of another. If Set B is a subset of Set A, the program should print "YES" followed by the sorted elements of Set B. If Set B is not a subset of Set A, the program should print "NO" followed by the average of all elements from both sets combined, rounded to two decimal places.

Implement a class Solution with the required method to perform the subset check using TreeSet in Java.

Input Format

The first line contains an integer n - the number of elements in Set A.

The second line contains n space-separated integers - the elements of Set A.

The third line contains an integer m - the number of elements in Set B.

The fourth line contains m space-separated integers - the elements of Set B.

Output Format

If Set B is a subset of Set A, print "YES" followed by the sorted values of Set B.

Otherwise, print "NO" followed by the average of all numbers in both sets (rounded to two decimal places).

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 5
1 2 3 4 5
3
2 3 5
Output: YES 2 3 5

Answer

```
import java.util.*;

// You are using Java

class Solution {
    public static void checkSubset(TreeSet<Integer> setA,TreeSet<Integer> setB,
int count, long sum){
        // Check subset
        if (setA.containsAll(setB)) {
            System.out.print("YES ");
            for (int x : setB) {
                System.out.print(x + " ");
            }
        }
        else {
            System.out.print("NO ");



            double avg = sum / count;
            System.out.printf("%.2f", avg);
        }


    }
}

class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        TreeSet<Integer> setA = new TreeSet<>();
        long sum = 0;
        for (int i = 0; i < n; i++) {
```

```
            int num = sc.nextInt();
            setA.add(num);
            sum += num;
        }
        int m = sc.nextInt();
        TreeSet<Integer> setB = new TreeSet<>();
        for (int i = 0; i < m; i++) {
            int num = sc.nextInt();
            setB.add(num);
            sum += num;
        }
        Solution.checkSubset(setA, setB, n + m, sum);
        sc.close();
    }
}
```

Status : Partially correct ***Marks : 7.5/10***

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
class Book {
    //Type your code here
    int ISBN;
    String title;
```

```
    String author;
    Book(int ISBN, String title,String author){
        this.ISBN=ISBN;
        this.title=title;
        this.author=author;
    }
    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;
    }
    public int hashCode(){
        return Integer.hashCode(ISBN);
    }
}

class Library {
    LinkedHashSet<Book> set;
    Library(){
        set=new LinkedHashSet<>();
    }
    public void addBook(int ISBN, String title,String author){
        Book b1=new Book(ISBN,title,author);
        set.add(b1);
    }
    public void removeBook(int isbn){
        Iterator<Book> itr=set.iterator();
       while(itr.hasNext()){
            Book b=itr.next();
            if(b.ISBN==isbn){
                set.remove(b);
                break;
            }
        }
    }

    public void displayBooks(){
        if(set.size()==0) {
            System.out.println("No books available");
            return;
        }
```

```
        Iterator<Book> itr=set.iterator();
        while(itr.hasNext()){
            Book b=itr.next();
            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 : Partially correct **Marks : 6/10**

4. 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 {
    int id;
    String name;
    String department;
    public Employee(int id, String name, String department) {
        this.id = id;
        this.name = name;
        this.department = department;
    }
    @Override


    public boolean equals(Object o) {
        if (o instanceof Employee) {
            return this.id == ((Employee) o).id;
        }
        return false;
    }
    @Override
    public int hashCode() {
        return Objects.hash(id);
    }
    @Override
    public String toString() {
        return "ID: " + id + ", Name: " + name + ", Department: " + department;
    }
}


class EmployeeDatabase {
    //Type your code here
    HashSet<Employee> employees = new HashSet<>();
    public void addEmployee(int id, String name, String department) {
        employees.add(new Employee(id, name, department));
    }
```

```
    public void displayEmployees() {
        for (Employee e : employees) {
            System.out.println(e);
        }
    }
    public boolean checkEmployee(int id) {
        for (Employee e : employees) {
            if (e.id == id) return true;
        }
        return false;
    }
}

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