**Problem Statement: Student Marks Management System**

**Problem Description:**

You are tasked with implementing a **Student Marks Management System**. The system will allow the user to input a list of students along with their corresponding marks, display the top-scoring students, and give the user the ability to query the marks of any student by their name. Additionally, after displaying a student's marks, the system should also display the student(s) with the highest marks.

**Requirements:**

1. **Input of Student Data:**
   * The system must accept the number of students.
   * For each student, the user must enter their name and marks.
2. **Top Scorer Identification:**
   * The system should identify the student(s) who achieved the highest marks.
3. **Student Marks Query:**
   * The system should allow the user to query the marks of any student by their name.
4. **Continuous Query Option:**
   * After displaying the list of students and their marks, the system should allow the user to repeatedly check the marks of any student by their name.
   * After displaying a student's marks, the program should also display the name of the student(s) who achieved the highest marks.
   * The user should be able to exit the query loop at any time.
5. **End the Program:**
   * After the user chooses not to check the marks anymore, the system should display the top-scoring student(s) again and then exit the program.
6. **Invalid Input Handling:**
   * If the user enters anything other than **'yes'** or **'no'**, the system should prompt the user to enter a valid input.

**Input Format:**

* The first line contains an integer numStudents (1 ≤ numStudents ≤ 100), which represents the number of students.
* For each student, the next two lines will contain:
  1. A string name (1 ≤ length(name) ≤ 100), representing the name of the student.
  2. An integer marks (0 ≤ marks ≤ 100), representing the marks of the student.

**Output Format:**

1. After all students have been entered, the system should print all student names and their marks in the following format:

Student Marks: {name1=marks1, name2=marks2, ...}

1. The system should also print the names of the top student(s) with the highest marks:

Top student(s) with highest marks (maxMarks): [name1, name2, ...]

1. The system should prompt the user if they want to check the marks of any student:

Do you want to check the marks of any student? (yes/no)

1. If the user answers yes, the system should ask for the name of the student and display their marks:

Enter the name of the student:

studentName's marks: marks

1. If the user answers no, the program will display the top student(s) again and exit the program:

Exiting program.

1. If the user enters an invalid input (anything other than yes or no), the system should prompt the user to enter a valid input:

Invalid input. Please type 'yes' or 'no'.

**Example 1:**

**Input:**

3

Aman

90

Shreya

91

Anuj

92

**Output:**

Student Marks: {Aman=90, Shreya=91, Anuj=92}

Top student(s) with highest marks (92): [Anuj]

Do you want to check the marks of any student? (yes/no):

yes

Enter the name of the student:

Shreya

Shreya's marks: 91

Top student(s) with highest marks (92): [Anuj]

Do you want to check the marks of any student? (yes/no):

yes

Enter the name of the student:

Aman

Aman's marks: 90

Top student(s) with highest marks (92): [Anuj]

Do you want to check the marks of any student? (yes/no):

no

Exiting program.

**Constraints:**

* 1 ≤ numStudents ≤ 100
* 1 ≤ length(name) ≤ 100
* 0 ≤ marks ≤ 100

**Notes:**

* The student names are case-sensitive.
* The program should handle multiple students having the same highest marks.
* The program should loop continuously for queries until the user decides to exit.

**Solution:**

import java.util.\*;

public class StudentMarks {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

Map<String, Integer> studMap = new HashMap<>();

System.out.println("Enter the number of students: ");

int numStudents = scanner.nextInt();

scanner.nextLine();

for (int i = 0; i < numStudents; i++) {

System.out.println("Enter student name: ");

String name = scanner.nextLine();

System.out.println("Enter marks for " + name + ": ");

int marks = scanner.nextInt();

scanner.nextLine();

studMap.put(name, marks);

}

System.out.println("\nStudent Marks: " + studMap);

int maxMarks = Collections.max(studMap.values());

Set<String> topStudents = new HashSet<>();

for (Map.Entry<String, Integer> entry : studMap.entrySet()) {

if (entry.getValue() == maxMarks) {

topStudents.add(entry.getKey());

}

}

System.out.println("\nTop student(s) with highest marks (" + maxMarks + "): " + topStudents);

while (true) {

System.out.println("\nDo you want to check the marks of any student? (yes/no): ");

String response = scanner.nextLine().toLowerCase();

if (response.equals("yes")) {

System.out.println("Enter the name of the student: ");

String studentName = scanner.nextLine();

if (studMap.containsKey(studentName)) {

System.out.println(studentName + "'s marks: " + studMap.get(studentName));

System.out.println("Top student(s) with highest marks (" + maxMarks + "): " + topStudents);

} else {

System.out.println("Student not found. Please try again.");

}

} else if (response.equals("no")) {

System.out.println("Exiting program.");

break;

} else {

System.out.println("Invalid input. Please type 'yes' or 'no'.");

}

}

System.out.println("\nTop student(s) with highest marks (" + maxMarks + "): " + topStudents);

}

}

**Complexities:**

**Time Complexity:**

* Worst-case time complexity: **O(numStudents + Q)** where numStudents is the number of students and Q is the number of queries.

**Space Complexity:**

* The space complexity is **O(numStudents)** due to the storage of student names and marks in the studMap, as well as the set storing the top-scoring students.