**Finding Common Students in Two Groups**

**Problem Statement:** Finding Common Students in Two Groups Using HashSet in C#

**Objective**: Write a C# program to find the common students enrolled in two different groups (e.g., students enrolled in Math and Science courses) using a `HashSet`.

**Description**:

You are tasked with developing a program for a school’s enrollment system. The system maintains lists of students enrolled in different courses. You need to identify students who are enrolled in both the Math course and the Science course. The student names are provided as two separate lists, and you must use a `HashSet` to efficiently find the common students (i.e., the intersection of the two groups). The program should display the names of the common students and the total count of such students. If there are no common students, the program should indicate that as well.

**Requirements**:

1. Use `HashSet<string>` to store the names of students in each group.

2. Perform the intersection operation to find students present in both groups.

3. Display each common student’s name.

4. Display the total number of common students.

5. Handle the case where there are no common students.

6. Ensure the solution is efficient and avoids duplicates (leverage `HashSet`’s built-in uniqueness).

**Input**:

- Two sets of student names:

- Math course students: { "Alice", "Bob", "Charlie", "David" }

- Science course students: { "Bob", "Charlie", "Eve", "Frank" }

**Output**:

- A list of students enrolled in both courses.

- The total count of common students.

- If no students are common, a message indicating this.

Example Output:

```

Students enrolled in both Math and Science:

Bob

Charlie

Total students in both courses: 2

```

**Constraints**:

- Student names are case-sensitive strings.

- The groups may have varying sizes, including empty sets.

- The solution must use `HashSet` for efficient intersection operations.

**Learning Goals:**

- Understand how to use `HashSet` in C#.

- Learn to perform set operations, specifically intersection, using `IntersectWith`.

- Practice handling collections and displaying results in a user-friendly format.