Assignment Case Study: Student Report Generator Refactoring

Problem Statement: Student Report Generator Refactoring

Objective:

You are provided with a legacy method GenerateStudentReport(Student student) that processes and prints a report for a student, including average grade calculation, letter grade assignment, attendance status, and output formatting.

The method is currently **monolithic**, performing **multiple responsibilities** in a single block of code.

Your task is to **refactor this method** into smaller, reusable, and testable methods following best practices such as:

- **Single Responsibility Principle**
- **Clean Code conventions**
- **Modularity and reusability**

Learning Goals:

- Apply **method decomposition** techniques
- Improve code **readability and testability**
- Practice **basic C# syntax** with lists, conditionals, and string formatting
- Reinforce understanding of **ref/out/optional parameters** (optional)

Requirements:

1. Refactor GenerateStudentReport(Student student) into at least **four helper methods**:
    - Validation logic
    - Average grade calculation
    - Letter grade evaluation
    - Attendance analysis
2. Create a clean, readable version of GenerateStudentReport that calls the helper methods in logical sequence.

3. Implement the following model:

```
class Student {
   public string Name { get; set; }
   public double[]  Grades ;
   public double AttendancePercentage { get; set; }
}
```

4. Keep the existing logic:
    - Average = mean of grades
    - Letter grade:
        - A: ≥ 90
        - B: ≥ 80
        - C: ≥ 70
        - D: ≥ 60
        - F: < 60
    - Attendance status:
        - "Satisfactory" if ≥ 75%
        - "Unsatisfactory" otherwise

Bonus Tasks (Optional but encouraged):

- Add a method to **detect failing grades** (less than 40)
- Add a **pass/fail result**:
    - Pass = average ≥ 60 **and** attendance ≥ 75%