# \*\*Python Script for Generating and Analyzing Student Grades\*\*

## \*\*Imports\*\*

The following libraries are utilized in this Python script:

- **statistics**:

- **Purpose**: This module provides functions to calculate mathematical statistics of numeric data.

- **Use in Script**: It is used to compute statistical metrics such as the mean and median of the student grades generated.

- **random**:

- **Purpose**: This module implements pseudo-random number generators for various distributions.

- **Use in Script**: It is employed to generate random grades for students within the specified range of 50 to 100.

## \*\*Functions\*\*

### \*\*1. generate\_student\_grades(num\_students)\*\*

- **Purpose**: This function generates a list of random integers representing student grades.

- **Parameters**:

- `num\_students` (int): Specifies the number of student grades to generate.

- **Return Value**:

- Returns a list of integers, each randomly selected between 50 and 100, representing the students' grades.

- **Mechanism**:

- The function utilizes the `random.randint(50, 100)` method to generate random grades and appends them to a list, which is returned after all grades are generated.

### \*\*2. calculate\_grade\_stats(grades)\*\*

- **Purpose**: This function computes various statistical metrics for a list of grades provided as input.

- **Parameters**:

- `grades` (list of int): A list containing the student grades for which statistics are to be calculated.

- **Return Value**:

- Returns a dictionary containing the following statistical metrics:

- `average`: The mean of the grades in the list.

- `median`: The median value of the grades.

- `highest`: The maximum grade.

- `lowest`: The minimum grade.

- **Mechanism**:

- The function utilizes methods from the `statistics` module to calculate the average, median, highest, and lowest values from the input list of grades, then returns these metrics in a dictionary format.

### \*\*3. main()\*\*

- **Purpose**: This is the main execution function of the script.

- **Mechanism**:

- The function orchestrates the program flow by executing the following steps:

1. Calls `generate\_student\_grades(10)` to create a list of 10 random student grades.

2. Passes the generated grades to `calculate\_grade\_stats` to compute statistical metrics.

3. Outputs the generated grades and the calculated statistics to the console in a readable format.

- **Execution**:

- The `main` function is invoked if the script is run directly, following standard Python script execution practices.

## \*\*Execution Mechanism\*\*

The script initiates with the `main` function, which serves as the orchestrator of the program's workflow. The following sequence outlines its execution process:

1. **Generate Grades**:

- Calls `generate\_student\_grades(10)` to produce a set of 10 random grades ranging from 50 to 100.

2. **Calculate Statistics**:

- Uses the list of grades produced to call `calculate\_grade\_stats(grades)`, obtaining statistical information.

3. **Print Results**:

- Outputs both the generated grades and computed statistics to the console for review.

This execution mechanism allows for a clear and efficient flow from grade generation to statistical analysis, ensuring ease of understanding and use.

## \*\*Technical Insights and Potential Improvements\*\*

- **Type Safety**:

- Consider implementing type hints for function parameters and return types to enhance code readability and provide better context.

- **Input Validation**:

- Implement validation for the `num\_students` parameter in `generate\_student\_grades` to prevent negative or zero values, which could lead to unexpected behavior.

- **Expand Statistical Analysis**:

- Enhance the `calculate\_grade\_stats` function to include additional statistical metrics, such as standard deviation or mode.

- **Flexible Grade Range**:

- Modify `generate\_student\_grades` to accept parameters for the minimum and maximum grade, providing greater flexibility in grade generation.

By addressing these recommendations, the script can be made more robust, user-friendly, and adaptable to various applications in educational settings.