Use JGrasp to design and implement the following programs (from the end of Chapter 8 in the textbook) and understand what they do.

**Required:** Use `assert` statement to validate program input values. See Chapter 3 slides for code examples. Apply this requirement to both lab exercises and assignment programs whenever constraints are applied to input values.

**Exercise 1:** Design and implement a Java program for programming exercise 8.1, page 305 (name it `SumArrayColumns`) as described in the problem statement. Write method `sumColumn()` as specified. To test this method, the main method of your program prompts the user to enter a 3-by-4 matrix and displays the sum of each column (by calling method `sumColumns()`). Design the main method of your program such that it allows the user to re-run the program with different inputs (i.e., use a loop). Document your code, and organize and space the outputs properly. Use escape characters and formatting objects when applicable.

**Exercise 2:** Design and implement a Java program for programming exercise 8.5, page 306 (name it `AddMatricies`) as described in the problem statement. Write method `addMatrix()` as specified. To test this method, the main method of your program prompts the user to enter two matrices that have the same dimensions, and then display their sum as shown in the sample run. Design the main method of your program such that it allows the user to re-run the program with different inputs (i.e., use a loop). Document your code, and organize and space the outputs properly. Use escape characters and formatting objects when applicable.

**Exercise 3:** Design and implement a Java program for programming exercise 8.13, page 310 (name it `LocateLargestElement`) as described in the problem statement. Write method `locateLargest()` as specified. Notice that this method returns a one-dimensional array that contains two elements. These two elements indicate the row and column indices of the largest element in the two-dimensional array. To test this method, the main method of your program prompts the user to enter a two-dimensional array and displays the location of the largest element in the array. Use the sample run to format your output. Design the main method of your program such that it allows the user to re-run the program with different inputs (i.e., use a loop). Document your code, and organize and space the outputs properly. Use escape characters and formatting objects when applicable.

**Exercise 4:** Design and implement a Java program for programming exercise 8.26, page 315 (name it `RowSorting`) as described in the problem statement. Write method `sortRows()` as specified (you may change the element type to integer). Notice that this method returns a new array, the original array still unchanged. See Chapter 7 for how to sort one-dimensional array. To test this method, the main method of your program prompts the user to enter a two-dimensional array and displays the original array followed by the row-sorted array as shown in the sample run. Design the main method of your program such that it allows the user to re-run the program with different inputs (i.e., use a loop). Document your code, and organize and space the outputs properly. Use escape characters and formatting objects when applicable.

**Instructions:**

1. Programs must be working correctly.
2. Programs must be completed and checked before working assignment #11.
3. Programs must be checked by the end of the designated lab session.