$X_1 = 2 \qquad X_2 = 5 \qquad X_3 = 0$

$R_1 \quad 2X_1 + 3X_2 - 4X_3 = 19 \qquad 2(2) + 3(5) - 4(0) = 19$

$R_2 \quad 4X_1 + 7X_2 + 1X_3 = 43 \qquad 4(2) + 7(5) + 1(0) = 43$

$R_3 \quad 6X_1 + 11X_2 + 2X_3 = 67 \qquad 6(2) + 11(5) + 2(0) = 67$

$$\begin{matrix} R_1 \\ R_2 \\ R_3 \end{matrix} \begin{bmatrix} 2 & 3 & -4 & 19 \\ 4 & 7 & 1 & 43 \\ 6 & 11 & 2 & 67 \end{bmatrix} \rightarrow \begin{bmatrix} 2 & 3 & -4 & 19 \\ 0 & 1 & 9 & 5 \\ 0 & 2 & 14 & 10 \end{bmatrix} \rightarrow \begin{bmatrix} 2 & 3 & -4 & 19 \\ 0 & 1 & 9 & 5 \\ 0 & 0 & -4 & 0 \end{bmatrix}$$

$R_2 = R_2 - \left(\frac{4}{2}\right) R_1$

$= 4 - (2)2 = 0$

$= 7 - (2)3 = 1$

$= 1 - (2)(-4) = 9$

$= 43 - (2)(19) = 5$

$R_3 = R_3 - \left(\frac{6}{2}\right) R_1$

$= 6 - (3)2 = 0$

$= 11 - (3)3 = 2$

$= 2 - (3)(-4) = 14$

$= 67 - (3)19 = 10$

$R_3 = R_3 - (2) R_2$

$= 2 - (2)1 = 0$

$= 14 - (2)9 = -4$

$= 10 - (2)5 = 0$

$2(x_1) + 3(x_2) - 4(x_3) = 19$

$(x_2) + 9(x_3) = 5$

$-4x_3 = 0 \rightarrow x_3 = 0$ *

$x_2 + 9(0) = 5 \rightarrow x_2 = 5$

$2(x_1) + 3(5) - 4(0) = 19$

$2x_1 = 4 \rightarrow x_1 = 2$ *

```java
public class GFA {

    private static double[][] problem1 = {
            // x = 1, y = 2, z = 3
            { 2, 3, -4, 19 }, // 1x + 2y + 3z = 14
            { 4, 7, 1, 43 }, // 1x - 1y + 1z = 2
            { 6, 11, 2, 67 } // 4x - 2y + 1z = 3
    };

    public static void solve(double[][] c, int row) {
        int cols = c.length + 1;
        double factor = c[row][row];
        for (int col = 0; col < cols; col++)
            c[row][col] /= factor;

        for (int row2 = 0; row2 < c.length; row2++)
            if (row2 != row) {
                factor = -c[row2][row];
                for (int col = 0; col < cols; col++)
                    c[row2][col] += factor * c[row][col];
            }
    }

    public static void solve(double[][] c) {
        for (int row = 0; row < c.length; row++)
            solve(c, row);
    }

    public static void print(double[][] c) {
        int cols = c.length + 1;
        for (int row = 0; row < c.length; row++) {
            for (int col = 0; col < cols; col++)
                System.out.printf("%5.1f ", c[row][col]);
            System.out.println();
        }
        System.out.println();
    }

    public static void printSolution(double[][] c) {
        int cols = c.length + 1;
        char variable = (char) ((c.length > 3) ? ('z' - (c.length - 1)) : 'x');
        System.out.println("Solution:\n");
        for (int row = 0; row < c.length; row++)
            System.out.printf("  %c = %1.1f\n", (char) variable++, c[row][cols - 1]);
        System.out.println();
    }

    public static void doProblem(double[][] problem, String description) {
        System.out.println("Original Equations:");
        print(problem);
        solve(problem);
        printSolution(problem);
    }

    public static void main(String[] args) {
        doProblem(problem1, "Problem 1 (from class)");

    }
}
```

```
Original Equations:
  2.0   3.0  -4.0  19.0
  4.0   7.0   1.0  43.0
  6.0  11.0   2.0  67.0

Solution:

  x = 2.0
  y = 5.0
  z = -0.0
```

Carmer Rule

$$\begin{bmatrix} 2 & 3 & -4 \\ 4 & 7 & 1 \\ 3 & 11 & 2 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 19 \\ 43 \\ 67 \end{bmatrix}$$

$$\det A = \begin{bmatrix} 2 & 3 & -4 \\ 4 & 7 & 1 \\ 6 & 11 & 2 \end{bmatrix} = -8$$

$$x = \begin{bmatrix} 19 & 3 & -4 \\ 43 & 7 & 1 \\ 67 & 11 & 2 \end{bmatrix} = \frac{-16}{-8} = 2$$

$$y = \begin{bmatrix} 2 & 19 & -4 \\ 4 & 43 & 1 \\ 6 & 67 & 2 \end{bmatrix} = \frac{-40}{-8} = 5$$

$$z = \begin{bmatrix} 2 & 3 & 19 \\ 4 & 7 & 43 \\ 6 & 11 & 67 \end{bmatrix} = \frac{0}{-6} = 0$$

```java
public class CramersRul {
    public static void main(String[] args) {
        double[][] coefficients = {
                { 2, 3, -4 },
                { 4, 7, 1 },
                { 6, 11, 2 }
        };

        double[] constants = { 19.0, 43.0, 67.0 };

        double determinant = calculateDeterminant(coefficients);

        if (determinant == 0) {
            System.out.println("The system has no unique solution.");
        } else {

            double[] solutions = new double[3];
            for (int i = 0; i < 3; i++) {
                double[][] modifiedMatrix = replaceColumn(coefficients, i, constants);
                solutions[i] = calculateDeterminant(modifiedMatrix) / determinant;
                System.out.printf("x%d = %.2f%n", i + 1, solutions[i]);
            }
        }
    }

    public static double calculateDeterminant(double[][] matrix) {
        return matrix[0][0] * (matrix[1][1] * matrix[2][2] - matrix[1][2] * matrix[2][1])
                - matrix[0][1] * (matrix[1][0] * matrix[2][2] - matrix[1][2] * matrix[2][0])
                + matrix[0][2] * (matrix[1][0] * matrix[2][1] - matrix[1][1] * matrix[2][0]);
    }

    public static double[][] replaceColumn(double[][] matrix, int columnIndex, double[] vector) {
        double[][] result = new double[matrix.length][matrix[0].length];
        for (int i = 0; i < matrix.length; i++) {
            for (int j = 0; j < matrix[0].length; j++) {
                if (j == columnIndex) {
                    result[i][j] = vector[i];
                } else {
                    result[i][j] = matrix[i][j];
                }
            }
        }
        return result;
    }
}
```

```
x1 = 2.00
x2 = 5.00
x3 = -0.00
```