1. (*Algebra: quadratic equations*) Design a class named **QuadraticEquation** for a quadratic equation
    $ax^2 + bx + c = 0$. The class contains:
    - The private data fields **a**, **b**, and **c** that represent three coefficients.
    - A constructor for the arguments for **a**, **b**, and **c**.
    - Three **get** methods for **a**, **b**, and **c**.
    - A method named **getDiscriminant()** that returns the discriminant, which is $b^2 - 4ac$.
    - The methods named **getRoot1()** and **getRoot2()** for returning the two roots of the equation using
      these formulas:

$$r_1 = \frac{-b + \sqrt{b^-4ac}}{2a} \text{ and } r_2 = \frac{-b - \sqrt{b^-4ac}}{2a}$$

These methods are useful only if the discriminant is nonnegative. Let these methods return **0** if the
discriminant is negative.

Draw the UML diagram for the class, and then implement the class. Write a test program that prompts the
user to enter values for a, b, and c and displays the result based on the discriminant. If the discriminant is
positive, display the two roots. If the discriminant is **0**, display the one root. Otherwise, display "The
equation has no roots."

2. (*Algebra: 2 × 2 linear equations*) Design a class named **LinearEquation** for a 2 × 2 system of
    equations.

$$ax + by = e$$

$$cx + dy = f$$

$$x = \frac{ed - bf}{ad - bc}$$

$$y = \frac{af - ec}{ad - bc}$$

The class contains:
- The private data fields **a**, **b**, **c**, **d**, **e**, and **f** with get methods.
- A constructor for the arguments for **a**, **b**, **c**, **d**, **e**, and **f**.
- Three **get** methods for **a**, **b**, **c**, **d**, **e**, and **f**.
- A method named **isSolvable()** that returns true if $ad - bc$ is not **0**.
- The methods named **getX()** and **getY()** that returns the solution for the equation.

Draw the UML diagram for the class, and then implement the class. Write a test program that prompts the
user to enter **a**, **b**, **c**, **d**, **e**, and **f** and displays the result. If $ad - bc$ is **0,** report that "The equation has no
solution."

3. (*Geometry: intersection*) Suppose two line segments intersect. The two endpoints for the first line
    segment are (**x1**, **y1**) and (**x2**, **y2**) and for the second line segment are (**x3**, **y3**) and (**x4**, **y4**). Write a
    program that prompts the user to enter these four endpoints and displays the intersecting point. (Hint
    use the class **LinearEquation** from problem 2).