QUEEN'S UNIVERSITY
SCHOOL OF COMPUTING

CISC212, FALL TERM, 2006
FINAL EXAMINATION
7pm to 10pm, 19 DECEMBER 2006, Jeffrey Hall 1st Floor

Instructor: Alan McLeod

If the instructor is unavailable in the examination room and if doubt exists as to the interpretation of any problem, the candidate is urged to submit with the answer paper a clear statement of any assumptions made.

Proctors are unable to respond to queries about the interpretation of exam questions. Do your best to answer exam questions as written.

Please write your answers in the boxes provided. Extra space is available on page 17 of the exam. The back of any page can be used for rough work. This exam refers exclusively to the use of the Java language. Comments are not required in the code you write. For full marks, code must be efficient as well as correct.

This is a closed book exam. No computers or calculators are allowed.

| Problem 1: | / 8 | Problem 4: | / 18 |
| Problem 2: | / 10 | Problem 5: | / 10 |
| Problem 3: | / 14 |

TOTAL: / 60
Problem 1) [4 marks for a) and 4 marks for b)]

a) Provide the output of each of the following println statements, in the box beside each statement. Remember that in Java, a float can only hold a maximum of 8 significant digits, and a float literal has a capital “F” appended to the number.

```java
float z = 20.0F;
float a = 1.0e-10F;
float x = 10.0F;
float b = 2.0e-10F;

System.out.println(b - 2 * a + z - x);
System.out.println(x + x - z - a);
System.out.println(x + x - a - z);
System.out.println(b - 2 * (a + z) - x);
```

b) The two equations shown below provide one of the roots of the quadratic equation: \( ax^2 + bx + c = 0 \). They are both algebraically correct equations. Circle the equation that you would use to calculate \( x_1 \) in a Java program in the case where the product of \( a \) times \( c \) is much smaller than \( b \).

\[
x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}
\]

\[
x_1 = \frac{-2c}{b + \sqrt{b^2 - 4ac}}
\]

Why did you choose this equation over the other? (One or two sentences only, please.)
Problem 3) [14 marks]

You must complete an object hierarchy by writing two classes: Square and Rectangle. You are given the classes: IllegalShapeException, AllSides and a class called TestShapes that shows how your objects must work. You are also given the interface: RequiredAccessorsInterface. The hierarchy is defined by:

- The class Rectangle must implement RequiredAccessorsInterface.
- The class Square must extend Rectangle.

As always, you are expected to use proper encapsulation practices. Write just the minimum amount of code required to satisfy the TestShapes class. **Do not** write any mutators, any other constructors than what is required, any other accessor than what is required, or any of the other standard methods, including clone(), equals(), compareTo() or toString().

An illegal dimension is less than or equal to zero. You will have space to write your two classes following this listing of the three supplied classes and the interface.

```java
public class IllegalShapeException extends Exception {
    public IllegalShapeException () {
        super("\nAttempt to create an illegal shape.\n");
    }
}

public class AllSides {
    public double side1, side2;
    public AllSides(double a) {
        side1 = a; side2 = 0;
    }
    public AllSides(double a, double b) {
        side1 = a; side2 = b;
    }
}

public interface RequiredAccessorsInterface {
    AllSides getSides ();
    double getArea();
}

public class Square extends Rectangle {
    public double side;
    public Square(double a) {
        side = a;
    }
}

public class Rectangle {
    public double side1, side2;
    public Rectangle(double a, double b) {
        side1 = a; side2 = b;
    }
}
```

Problem 3, Cont.)

public class TestShapes {

    public static void testAccessors (RequiredAccessorsInterface shape) {
        System.out.println("\nTesting: " + shape.getClass().getName());
        System.out.println("Area = " + shape.getArea());

        String s = "Side1 = " + shape.getSides().side1;
        if (shape.getSides().side2 > 0)
            s += ", Side2 = " + shape.getSides().side2;
        System.out.println(s);
    } // end testAccessors

    public static void main(String[] args) {

        Square illegalSquare = null, legalSquare = null;
        Rectangle illegalRectangle = null, legalRectangle = null;

        // Square
        try {
            illegalSquare = new Square(-2.0);
        } catch (IllegalShapeException e) {
            System.out.println(e.getMessage());
        }
        try {
            legalSquare = new Square(2.0);
        } catch (IllegalShapeException e) {
            System.out.println(e.getMessage());
        }
        testAccessors(legalSquare);

        // Rectangle
        try {
            illegalRectangle = new Rectangle(-2.0, 3.0);
        } catch (IllegalShapeException e) {
            System.out.println(e.getMessage());
        }
        try {
            legalRectangle = new Rectangle(2.0, 3.0);
        } catch (IllegalShapeException e) {
            System.out.println(e.getMessage());
        }
        testAccessors(legalRectangle);

        } // end main

    } // end TestShapes
Problem 3, Cont.)

/* Output from TestShapes:

Attempt to create an illegal shape.

Testing: Square
Area = 4.0
Side1 = 2.0

Attempt to create an illegal shape.

Testing: Rectangle
Area = 6.0
Side1 = 2.0, Side2 = 3.0
*/
Problem 5) [10 marks]

Complete the following GUI class. You must add code to the constructor to finish the layout of the two components – a label and a command button. Then you must complete the listener class so that the button causes the label to switch between two messages. The window looks like:

Clicking on the “Switch” button switches the message on each click.

Remember that a button’s listener class must implement the ActionListener interface which will force it to have a concrete implementation of the method:

```
public void actionPerformed(ActionEvent e)
```

To set the text of a label, you invoke the setText() mutator method. To add a component to a panel or a JFrame, you invoke the add() method.

The listing of the class you must complete begins on the next page. Add code where indicated only, and do not change any of the supplied code.

Note that you cannot get this layout by adding the button and label components directly to the frame; you must use a Container component instead (like a JPanel...).
Problem 5, Cont.)

import javax.swing.JFrame;
import javax.swing.JButton;
import javax.swing.JPanel;
import javax.swing.JLabel;
import java.awt.Font;
import java.awt.BorderLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class GUIExamProblemWindow extends JFrame {
}

// COMPLETE THE CONSTRUCTOR HERE:
Problem 5, Cont.)

    // WRITE THE LISTENER CLASS HERE:

} // end GUIExamProblemWindow class