NAME: ______________________________________________

CU ID: _______________________________________________

Section TA/time: _______________________________________

You have one and a half hours to do this exam.
All programs in this exam must be written in Java.
The exam consists of 12 pages. Make sure you have all of them.

********************************************************************************

<table>
<thead>
<tr>
<th>Problem</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>30</td>
</tr>
<tr>
<td>2</td>
<td>40</td>
</tr>
<tr>
<td>3</td>
<td>30</td>
</tr>
</tbody>
</table>

Total: 100
Java Nagila, our novice programmer, has just been learning about fractions. Each fraction is written as a ratio of two integers: a numerator and a non-zero denominator. For example, $\frac{1}{2}$ is a fraction, there 1 is the numerator and 2 is the denominator. This fraction can also be represented by the real number 0.5.

Ms. Nagila has written some code which is given below. Answer the questions from (a) to (f) in the space provided in the code.

// (a) (5 points) Draw the complete inheritance hierarchy for the classes and interfaces in this problem.
public interface IReal { double getReal(); }

public interface IFraction extends IReal {
    int getNumerator();
    int getDenominator();
}

public class Real implements IReal {
    protected double rVal;

    public Real(double r) { rVal = r; }
    public double getReal() { return rVal; }

    // (b) (5 points) Write the equals() method which returns true if the
    // values in the two Reals being compared are equal.
    public boolean equals(Object obj) {
        if (!(obj instanceof Real)) return false;
        return rVal == ((Real)obj).rVal;
    }

    // Return the real value as a String
    public String toString() {
        return String.valueOf(rVal);
    }

    // Make a new Real object based on the argument r.
    public static IReal makeCopy(IReal r) {
        return new Real(r.getReal());
    }
}

final public class Fraction extends Real implements IFraction {
    private int numerator;
    private int denominator;

    // (c) (5 points) Write the constructor that initializes the state
    // correctly. It should make sure that the corresponding real value is
    // stored in its inherited state.
    public Fraction(int n, int d) {
        super((double)n/d);
        numerator = n;
        denominator = d;
    }

    public int getNumerator() { return numerator; }
    public int getDenominator() { return denominator; }

    // (d) (8 points) Write the equals() method which returns true if the
    // numerator and denominator values in the two Fractions
    // being compared are equal.
    public boolean equals(Object obj) {
        if (!(obj instanceof Fraction)) return false;
        return (numerator == ((Fraction)obj).numerator) &&
               (denominator == ((Fraction)obj).denominator);
    }

    // (e) (3 points) Write the toString() method which returns a String with
public String toString() {
    return numerator + "/" + denominator;
}

public static IFraction makeCopy(IFraction q) {
    return new Fraction(q.getNumerator(), q.getDenominator());
} // end of Fraction
This page is intentionally left blank.
Problem 2 (40 points)

Java Nagila has written a class called ClientA that uses the interfaces and classes defined in Problem 1. It is one of those days, and she is wondering if you can help her by answering the questions from (a) to (h) in the space provided in the code.

```java
public class ClientA {
    public static void main(String args[]) {
        // (a) (3 points) Declare an array reference named numbers which
        // can have elements of type IReal.
        IReal[] numbers;

        // (b) (3 points) Create an appropriate array of 5 elements denoted
        // by the array reference numbers.
        numbers = new IReal[5]; // numbers = new Real[5];

        // (c) (5 points) Initialize the elements of the array denoted by the
        // reference numbers.
        // Starting with the 1st element, the array elements denote objects of
        // the appropriate type having the following values respectively:
        // 0.5, 1/4, 0.75, 1/2, 3/4.
        numbers[0] = new Real(0.5);
        numbers[1] = new Fraction(1, 4);
        numbers[2] = new Real(0.75);
        numbers[3] = new Fraction(1, 2);
        numbers[4] = new Fraction(3, 4);

        // (d) (5 points) Write a for-loop that prints out the values of the
        // elements of the array numbers on the terminal.
        for (int i = 0; i < numbers.length; i++)
            System.out.println(numbers[i]);

        // (e) (10 points) Write a for-loop that prints out the values of the
        // elements of the array numbers, which have the denominator value 4,
        // on the terminal.
        for (int i = 0; i < numbers.length; i++) {
            IReal realRef = numbers[i];
            if ((realRef instanceof IFraction) &&
                    ((IFraction)realRef).getDenominator() == 4)
                System.out.println(realRef);
        }

        // (f) (5 points) What will the method checkItOut print when called
        // as follows:
        // true
        // false
        // false
        // true
        // false
        checkItOut(new Real(0.5), numbers); // Method is given below.
        System.out.println();
    }
}
```

The `checkItOut` method is not provided in the text. It might be defined elsewhere in the code. To complete Problem 2, you need to write the `checkItOut` method that checks if the first element of the `numbers` array is a `Real` object with value 0.5.
(g) (5 points) What will the method checkItOut print when called as follows:
false
false
false
true
false
checkItOut(new Fraction(1,2), numbers);  // Method is given below.
System.out.println();

(h) (4 points) What will method justDoIt print when called as follows:
0.5
1/2
justDoIt(new Fraction(1,2));  // Method is given below.
System.out.println();

} // end of ClientA
This page is intentionally left blank.
Problem 3 (30 points)

The circumference of a circle is given by the following formula when the radius is known:
\[ \text{Circumference} = 2 \times \pi \times \text{Radius}. \]

The radius of a circle is given by the following formula when the circumference is known:
\[ \text{Radius} = \text{Circumference} / (2 \times \pi) \]

In this small GUI-based application, the user can type in the radius of a circle and the circumference of the circle is calculated and displayed. If the user types in the circumference of a circle, then the radius of the circle is calculated and displayed.

The value of \( \pi \) is given by the constant \text{Math.PI}. A string can be parsed (i.e. converted) to a double value using the \text{Double.parseDouble()} method. A double value can be converted to a string using the \text{String.valueOf()} method. For this application, assume that the user always types in legal values. The \text{getText()} and \text{setText()} method can be used to get and set the text in a TextField.

The GUI is shown in Figure 4a and the containment hierarchy is shown in Figure 4b. The skeleton code is provided below. Write the method \text{createGUI()} which creates the GUI according to the containment hierarchy in Figure 4b, and the method \text{addListeners()} which creates and adds the relevant listeners to the relevant sources. The user should be able to terminate the program when the “close-box” of the top-level window is clicked.

Table 4 provides information about sources, events and listeners. A TextField generates an \text{ActionEvent} when the user hits the ENTER key. A Window generates a \text{WindowEvent}. The Window calls the \text{windowClosing()} method of the \text{WindowListener} interface when the user clicks the close-box of the window. Note also that the class \text{WindowAdapter} implements the \text{WindowListener} interface, and can facilitate implementing listeners of \text{WindowEvent}.

![Figure 4a](image1.png)

![Figure 4b](image2.png)

Figure 4
import java.io.*;
import java.awt.*;
import java.awt.event.*;

/**
 * Circumference of a Circle
 */
public class CircumferenceOfCircle extends Frame {

    // Label for the circumference formula
    private Label formulaLabel;
    private Panel formulaPanel;
    // Text field for the circumference value
    private TextField circumTF;
    // Text field for the radius value
    private TextField radiusTF;

    public CircumferenceOfCircle() {
        super("Circumference Of Circle");
        // Create the GUI
        createGUI();
        // Add the listeners to the sources
        addListeners();
        // Pack and show the window
        pack();
        setVisible(true);
    }

    // Method to terminate the program
    private void terminate() {
        dispose();
        System.exit(0);
    }

    // The main method
    public static void main (String[] args) {
        new CircumferenceOfCircle();
    }
}

Table 4

// Source code
import java.io.*;
import java.awt.*;
import java.awt.event.*;
/**
 * Circumference of a Circle
 */
public class CircumferenceOfCircle extends Frame {

    // Label for the circumference formula
    private Label formulaLabel;
    private Panel formulaPanel;
    // Text field for the circumference value
    private TextField circumTF;
    // Text field for the radius value
    private TextField radiusTF;

    public CircumferenceOfCircle() {
        super("Circumference Of Circle");
        // Create the GUI
        createGUI();
        // Add the listeners to the sources
        addListeners();
        // Pack and show the window
        pack();
        setVisible(true);
    }

    // Method to terminate the program
    private void terminate() {
        dispose();
        System.exit(0);
    }

    // The main method
    public static void main (String[] args) {
        new CircumferenceOfCircle();
    }
}
// (a) (15 points) Write method createGUI to set up the GUI.
private void createGUI() {

    formulaLabel = new Label("Circumference = 2 * PI * Radius", Label.CENTER);

    formulaPanel = new Panel();
    circumTF = new TextField(10);
    radiusTF = new TextField(10);
    formulaPanel.add(circumTF);
    formulaPanel.add(radiusTF);

    add(formulaLabel, BorderLayout.NORTH);
    add(formulaPanel, BorderLayout.CENTER);
}

// (b)(15 points) Write method addListeners to add listeners to
// the sources.
private void addListeners() {

circumTF.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        double circumference = Double.parseDouble(circumTF.getText());
        double radius = circumference / (2 * Math.PI);
        radiusTF.setText(String.valueOf(radius));
    }
});

radiusTF.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent evt) {
        double radius = Double.parseDouble(radiusTF.getText());
        double circumference = 2 * Math.PI * radius;
        circumTF.setText(String.valueOf(circumference));
    }
});

addWindowListener(new WindowAdapter() {
    public void windowClosing(WindowEvent evt) {
        terminate();
    }
});

} // end of CircumferenceOfCircle
This page is intentionally left blank.