CS/120 Sample Exam #1 -- Riley

Please show all of your work.

1. For each part below write the term, symbols, or phrase from class that best fits the description. (Each part is worth 2 points.)

a) Every program has syntax, which refers to the form of the code, and ______, which refers to the meaning of the code.

2. For each of the following code segments give the value that is assigned to the mysteryChip variable by the time that the code finishes executing.

a) int mysteryChip;
   mysteryChip = 21;
   mysteryChip = mysteryChip * 3;
   mysteryChip--;

b) double mysteryChip;
   mysteryChip = 6.5 - 8 / 3;

c) int mysteryChip;
   mysteryChip = (int)(7.5 * 3 + 1);

d) Oval oreoChip = new Oval(1, 2, 3, 4);
   int mysteryChip = (oreoChip.getWidth() + oreoChip.getHeight()) % 5;

e) int mysteryChip = (int)( (double)1.7 + (int)4.8 );
Circle each instruction below that is correct (i.e., does not result in a syntax error) when located in place of the INSTRUCTION GOES HERE line.

- fritos = 100;
- cheetos = 100;
- popcorn = 100;
- pretzel = 100;
- pretzel = (double)100;
- pretzel = fritos;
- pretzel = cheetos;
- pretzel = popcorn;

```
private int combiner(int j, int k, int m) {
    return j + k + m;
}
```

4. Use the combiner method above to complete Parts a and b.

6 pts  a) For every line below, circle the line if it is a syntactically correct Java statement when included in the body of another method of the same class as the combiner method.

```
int stuff = combiner(0, 1, 3);
int differentStuff = combiner(0.0, 100.0, 200.0);
int lessStuff = combiner(4, 3, 2, 1);
double doubleStuff = combiner(1, 2, 3);
int tuffStuff = combiner(10, 20, 30) + combiner(30, 20, 10);
```

6 pts  b) What value is assigned to stuffMore and stuffIt when the following code executes?

```
int stuffMore = combiner(10, 0, 2) - combiner(7, 1, 8);
int stuffIt = combiner(100, 0, 2 * combiner(1, 2, 3));
```
5. In the space below complete the \texttt{jthChar} method. This method returns the character from position \texttt{j} of the String \texttt{s}. You may assume that \texttt{j} is less than \texttt{s.length()}.  
\begin{verbatim}
    private char jthChar( int j, String s ) {

7. In the space below write a \texttt{box} method that returns a graphical object consisting of a square centered within another square. This method has a two double parameters and a Color parameter. The first parameter is the side length of an outer filled square. The second parameter is the distance between the outer edge of the outer square and an inner filled square. The third parameter is the color of the outer square. The inner square is black. You may assume that the value of the second argument is less than half of the first argument.
7. Write a non-void Java method that returns a random integer from 2 through 10.

8. In the space provided draw a picture of the image that appears within the JFrame, called win, as a result of the following Java code execution. (You may assume that all classes from java.awt and javax.swing have been imported by the class.)

```java
JFrame win;
Oval round, rounder;
Container notRound;
win = new JFrame("window");
win.setBounds(10, 10, 600, 500);
win.setLayout(null);
win.setVisible(true);

notRound = new Container();
notRound.setBounds(400, 0, 200, 100);
win.add(notRound, 0);
round = new Oval(2, 2, 196, 196);
round.setBackground(Color.lightGray);
win.add(round, 0);
rounder = new Oval(-100, 0, 200, 200);
rounder.setBackground(Color.black);
rounder.add(round, 0);
notRound.add(rounder, 0);
win.repaint();
```
9. On the next page complete **all** necessary code to accomplish the following. Whenever possible declare your variables locally. (Recall that an **ThreeButtonFrame** is a **JFrame** that is 600 pixels wide and 500 pixels tall.)

- As the program begins, there is one **ThreeButtonFrame** window colored white. This window has a vertical red line drawn from top to bottom through the window’s middle and it has a green filled circle (a dot) that with a diameter of 20 that is just touching the upper left corner of the window.

- Clicking **left button** causes the previous line to disappear and be replaced by a red line drawn from the upper left corner of the window to a point with identical x and y coordinates calculated as ten times the number of times that the MID button has been clicked during this program execution.

- The window does not visibly change when the **mid button** is clicked.

- Every time the **right button** is clicked the **ThreeButtonFrame**'s color changes. If it was white, it becomes black; and if it was black, it becomes white.
import java.awt.*;
import javax.swing.*;

public class Driver extends ThreeButtons {

    public Driver() {

        win = new ThreeButtonFrame("title");
    }

    public void leftAction() {

    }

    public void midAction() {

    }

    public void rightAction() {

    }
}
### Math

<table>
<thead>
<tr>
<th>double final</th>
<th>PI</th>
</tr>
</thead>
</table>

#### `abs`
- int abs(int)
- double abs(double)
- double cos(double)
- double sin(double)
- double tan(double)
- double pow(double, double)
- double random()
- long round(double)
- double sqrt(double)
- double toDegrees(double)
- double toRadians(double)

#### `double`
- cos(double)
- sin(double)
- tan(double)
- sqrt(double)
- toDegrees(double)
- toRadians(double)

### Line

| - int x |
| - int y |
| - int width |
| - int height |
| - Color color |

#### Constructor
- Line(int, int, int, int)

#### Update
- void setBackground(java.awt.Color)
- void repaint()

#### Query
- java.awt.Color setBackground()
### javax.swing.JFrame

- int x
- int y
- int width
- int height
- Color backColor

**<constructor>**
+ JFrame( String )

**<update>**
+ void add( java.awt.Component, int )
+ void remove( java.awt.Component )
+ void setBackground( java.awt.Color )
+ void setBounds( int, int, int, int )
+ void setLocation( int, int )
+ void setLayout( java.awt.LayoutManager )
+ void setVisible( boolean )
+ void setSize( int, int )
+ void repaint()

**<query>**
+ java.awt.Color getBackground()
+ int getX()
+ int getY()
+ int getHeight()
+ int getWidth()
+ . . .

### java.awt.Container

- int x
- int y
- int width
- int height

**<constructor>**
+ Container()

**<update>**
+ void add( java.awt.Component, int )
+ void remove( java.awt.Component )
+ void setBackground( java.awt.Color )
+ void setBounds( int, int, int, int )
+ void setLocation( int, int, int, int )
+ void setLayout( java.awt.LayoutManager )
+ void setVisible( boolean )
+ void setSize( int, int )
+ void repaint()

**<query>**
+ int getX()
+ int getY()
+ int getHeight()
+ int getWidth()
+ . . .

### Rectangle

- int x
- int y
- int width
- int height
- Color color

**<constructor>**
+ Rectangle( int, int, int, int )

**<update>**
+ void add( java.awt.Component, int )
+ void remove( java.awt.Component )
+ void setBackground( java.awt.Color )
+ void setBounds( int, int, int, int )
+ void setLocation( int, int )
+ void setSize( int, int )
+ void repaint()

**<query>**
+ java.awt.Color getBackground()
+ int getX()
+ int getY()
+ int getWidth()
+ int getHeight()
+ . . .

### Oval

- int x
- int y
- int width
- int height
- Color color

**<constructor>**
+ Oval( int, int, int, int )

**<update>**
+ void add( java.awt.Component, int )
+ void remove( java.awt.Component )
+ void setBackground( java.awt.Color )
+ void setBounds( int, int, int, int )
+ void setLocation( int, int, int, int )
+ void setSize( int, int, int, int )
+ void repaint()

**<query>**
+ java.awt.Color getBackground()
+ int getX()
+ int getY()
+ int getWidth()
+ int getHeight()
+ . . .