**Java Final Part 1**
Kurt Medley

1.a if (x && y == 0) { x = 1; y = 1; } Syntax error in "if" statement. Must compare x and y separately; so if (x == 0 && y == 0) { x = 1; y = ; }
b. if (1 <= x <= 10) Syntax error. must compare integer values with x separately. (x >= 1 && x <= 10)
c. if (!s.equals("nickels") || !s.equals("pennies") || !s.equals("dimes") || !s.equals("quarters"))
Statement should read (!(s.equals("nickels") || !(s.equals("pennies") || !(s.equals("dimes") || !(s.equals("quarters"))
where the not "!" operator should precede each condition within parenthesis.
d. if (input.equalsIgnoreCase("N") || "NO") should read if (input.equalsIgnoreCase("N") || input.equals("NO"))

2. r.equals(s) is meant to determine equality between strings whereas r == s tests whether two string variables refer to the identical string object.

3. The difficulty in comparing floating-point numbers, especially ones that represent numbers 2^(-1), 2^(-2), --, is that roundoff errors can represent different numbers once calculated/called later in life.

```
public int compareInt(int input) {
        n = input;
        if (n == 10) {
                S.O.PL(n + "= 10); }
        else{
                S.O.PL(n + "!= 10); }
        }

public double doubleCompare(double input) {
        final double EPSILON = 1E-14;
        n = input;
        if (Math.abs(n - 10) <= EPSILON) {
                S.O.PL(n + "is approximately equal to 10"); }
        else{
                S.O.PL(n + "is not approximately equal to 10"); }
        }
```

4. The building parameters must include a call to the Rectangle class. R has not been initialized. When testing for null, the object must have already been constructed and initialized. Also, the == operator must be used instead of the ".equals".

5.
```
        public class bridgeconverter {
                private String bridgename;
                private double bridgelength;

                public bridgeconverter() {
                        bridgename = "";
                        bridgelength = 0;

                public double convertFtToM(double ft)


        boolean bridgeNameOk(String s);
        boolean bridgeLengthOK(double ft);
        double convertFtToM(double ft);
```

6.
```
        int s = 0;
        int i = 1;
        while (i <= 10) {
                s = s + i;
                i ++; }
```
7.

8.
a) True
b) True
c) False
d) False
e) True
f) False
g) False
h) True

9. Like the Fibonacci program in chapter 6, this swap method works by creating a local variable temp that holds the initial value of a, changing value of a to b, and then changing the value of b to temp (which was originally a's value).

10.