CSE 142, Autumn 2006
Sample Midterm Exam #1

1. Expressions (10 points)

For each expression in the left-hand column, indicate its value in the right-hand column. Be sure to list a constant of appropriate type (e.g., `7.0` rather than `7` for a `double`, `String`s in quotes).

Expression	Value
`3 * 4 + 5 * 6 + 7 * -2`	_____
`1.5 * 2.0 + (5.5 / 2) + 5 / 4`	_____
`23 % 5 + 31 / 4 % 3 - 17 % (16 % 10)`	_____
`"1" + 2 + 3 + "4" + 5 * 6 + "7" + (8 + 9)`	_____
`345 / 10 / 3 * 55 / 5 / 6 + 10 / (5 / 2.0)`	_____

2. Parameters (20 points)

At the bottom of the page, write the output produced by the following program.

```
public class ParameterMystery {
    public static void main(String[] args) {
        String x = "java";
        String y = "tyler";
        String z = "tv";
        String rugby = "hamburger";
        String java = "donnie";

        hamburger(x, y, z);
        hamburger(z, x, y);
        hamburger("rugby", z, java);
        hamburger(y, rugby, "x");
        hamburger(y, y, "java");
    }

    public static void hamburger(String y, String z, String x) {
        System.out.println(z + " and " + x + " like " + y);
    }
}
```

3. While Loop Simulation, 15 points. For each call of the method below, write the output that is printed:

```
public static void mystery(int i, int j) {
    while (i != 0 && j != 0) {
        i = i / j;
        j = (j - 1) / 2;
        System.out.print(i + " " + j + " ");
    }
    System.out.println(i);
}
```

Method Call	Output
mystery(5, 0)	_____
mystery(3, 2)	_____
mystery(16, 5)	_____
mystery(80, 9)	_____
mystery(1600, 40)	_____

4. Assertions, 15 points. For the following method, identify each of the three assertions in the table below as being either ALWAYS true, NEVER true or SOMETIMES true / sometimes false at each labeled point in the code.

```
public static int mystery(int x) {
    int y = 1;
    int z = 0;

    // Point A
    while (y <= x) {
        // Point B
        y = y * 10;
        z++;

        // Point C
    }

    // Point D
    z--;

    // Point E
    return z;
}
```

	y > x	z < 0	z > 0
Point A			
Point B			
Point C			
Point D			
Point E			

5. Programming, 15 points.

Write a static method named `hasMidpoint` that accepts three integers as parameters and returns `true` if one of the integers is the midpoint between the other two integers; that is, if one integer is exactly halfway between them. Your method should return `false` if no such midpoint relationship exists.

Note that the three integers could be passed in any order; the midpoint could be the first, second, or third integer, so you will have to check all of these cases.

Calls such as the following should return `true` :

```
hasMidpoint(4, 6, 8)
hasMidpoint(2, 10, 6)
hasMidpoint(8, 8, 8)
hasMidpoint(25, 10, -5)
```

Calls such as the following should return `false` :

```
hasMidpoint(3, 1, 3)
hasMidpoint(1, 3, 1)
hasMidpoint(21, 9, 58)
hasMidpoint(2, 8, 16)
```

6. Programming (15 points)

Write a static method named `countEvenDigits` that accepts an integer as its parameter and returns the number of even-valued digits in that number. An even-valued digit is either 0, 2, 4, 6, or 8.

For example, the number 8546587 has four even digits (the two 8s, the 4, and the 6), so the call `countEvenDigits(8346387)` should return 4.

You may assume that the value passed to your method is non-negative.

7. Programming (10 points)

Write a method `printStripped` that accepts a `String` as a parameter and that prints a complete line of output with any comments stripped from the string. Comments are defined to be characters enclosed in the characters < and >. One way to think of this is that text is "normal" until you encounter a < character. From that point on, the text is considered a comment until you encounter a > character, at which point you return to normal text. This definition allows for < inside a comment and > outside a comment. You may assume that there are no unclosed comments in any given line.

For example, the following sequence of calls:

```
printStripped("this is plain text");
printStripped("this has a normal comment <right here> to be removed");
printStripped("this has multiple less-than in a comment <<<<<see?>");
printStripped("this > has <comment>greater-than outside a comment >>");
printStripped("<hi>this has <foo> multiple <<bar>> comments<baz><>.");
```

should produce the following output:

```
this is plain text
this has a normal comment to be removed
this has multiple less-than in a comment
this > has greater-than outside a comment >>
this has multiple > comments.
```