CSE 142 SUMMER 2007 MIDTERM

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"). If the expression is illegal, then write "error".

Expression	Value
`6 - 3 * 2 + 1 + "6"`	_____
`6 % 4 % 9 + 5 - 13 % 5`	_____
`2 + 4 * 2 / 3 * 3 + 1`	_____
`1 * 2 + 3 * 4 + 5 + 6`	_____
`0.25 + 3 / 4 / 3 + 0.75`	_____

2. Parameter Mystery (20 points)

At the bottom of the page, write the output produced by the following program, as it would appear on the console.

```
public class ParameterMystery {
    public static void main(String[] args) {
        String green = "i";
        String am = "green";
        String ham = "sam";
        String i = "eggs";
        String eggs = "am";
        String sam = "ham";

        mystery(sam, i, am);
        mystery(ham, green, eggs);
        mystery(green, eggs, ham);
        mystery(i, sam, "green");
        mystery(i, "am", sam);
    }

    public static void mystery(String eggs, String ham, String green) {
        System.out.println("I do not like " + green + " " + eggs + " and " + ham);
    }
}
```

3. While Loop Simulation (15 points)

For each call below to the following method, write the output that is printed, as it would appear on the console:

```
public static void mystery(int a, int b) {
    while (b > 0) {
        a = a - b;
        b = a - b;

        System.out.print(a + " " + b + " ");
    }
}
```

Method Call	Output
`mystery(16, 4);`	_____
`mystery(11, -7);`	_____
`mystery(25, 10);`	_____
`mystery(-6, 10);`	_____
`mystery(10, 3);`	_____

4. Assertions (15 points)

For each of the five points labeled by comments, identify each of the following assertions as being either always true, never true or sometimes true / sometimes false.

```
public static void mystery(int a) {
    int b = 0;

    // Point A
    while (a > 0) {
        // Point B
        if (a % 2 == 0) {
            b++;
            // Point C
        } else {
            a--;
        }

        // Point D
        a /= 2;
    }

    // Point E
}
```

	a % 2 == 0	a > 0	b > 0
Point A			
Point B			
Point C			
Point D			
Point E			

5. Conditionals (9 points)

Consider the following piece of code where `a` and `b` are Boolean variables:

```
if (a && b) {
    System.out.print("Hi");
} else if (b) {
    System.out.println(" there!");
}
```

(a) Under what conditions will the code print out only " `there!`"? Explain your answer using 20 words or less.

(b) Under what conditions will the code print out "`Hi there!`"? Explain your answer using 20 words or less.

6. Programming (15 points)

Morse code is a translation from letters to a series of dots and dashes. For example, the first five letters are translated as follows:

A = .- B = -… C = -.-. D = -.. E = .

Write a static method named `printInMorse` that accepts a word as a parameter and prints out the Morse version of the word. Assume words only consist of the above letters. Case does not matter. Letters in Morse code are separated by spaces (it is OK to have a space after the last Morse letter). If it helps, you can write helper methods (extra methods that get used by the primary method).

Here are some example calls to the method and their expected output:

Call	Output
`printInMorse("bead")`	`-... . .- -..`
`printInMorse("Dad")`	`-.. .- -..`
`printInMorse("aBcDe")`	`.- -... -.-. -.. .`
`printInMorse("cab")`	`-.-. .- -...`

7. Programming (15 points)

Write a static method named `battleToTheDeath` that simulates a fight between two players. It accepts two integer parameters representing the minimum and maximum amount of damage a player can inflict upon the other player. The "health" of a player is represented by a number initially set to 100. Each player randomly attacks the other, subtracting damage from the attacked player's health, until one of the player's health falls below 1. The damage is a random number between the minimum damage value and the maximum damage value (inclusive).

You are to reproduce the format of the sample output below. Here is a sample run for the call `battleToTheDeath(20,40)`:

Call	Output

`battleToTheDeath(20,40);`	`DEATH MATCH!!!`
	`Player 2 attacks! -40 damage.`
	`Player 1 attacks! -31 damage.`
	`Player 1 attacks! -32 damage.`
	`Player 2 attacks! -23 damage.`
	`Player 1 attacks! -35 damage.`
	`Player 2 attacks! -30 damage.`
	`Player 1 attacks! -28 damage.`
	`Player 1 wins!`

You may assume that the parameters passed to your method will be non-negative and the maximum damage value (the second parameter) will always be greater than the minimum damage value (the first parameter).