CSE 142, Autumn 2006
Sample Midterm Exam #2

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
`8 + 5 * 3 / 2`	_____
`1.5 * 4 * 7 / 8 + 3.4`	_____
`73 % 10 - 6 % 10 + 28 % 3`	_____
`4 + 1 + 9 + "." + (-3 + 10) + 11 / 3`	_____
`3 / 14 / 7 / (1.0 * 2) + 10 / 6`	_____

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 = "happy";
        String y = "pumpkin";
        String z = "orange";
        String pumpkin = "sleepy";
        String orange = "vampire";

        orange(y, x, z);
        orange(x, z, y);
        orange(pumpkin, z, "y");
        z = ”green”;
        orange("x", "pumpkin", z);
        orange(y, z, orange);
    }

    public static void orange(String z, String y, String x) {
        x = x + “!”;
        System.out.println(y + " and " + z + " were " + x);
    }
}
```

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

```
public static int mystery(int i, int j) {
    int k = 0;
    while (i > j) {
        i = i - j;
        k += (i-1);
    }
    return k;
}
```

Method Call	Value returned
`mystery(2, 9)`	_____
`mystery(5, 1)`	_____
`mystery(38, 5)`	_____
`mystery(5, 5)`	_____
`mystery(40, 10)`	_____

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(Scanner console) {
    int y = 0;
    int z = 1;
    int next = console.nextInt();

    // Point A
    while (next >= 0) {
        // Point B
        if (y > z) {
            // Point C
            z = y;
        }
        y++;
        next = console.nextInt();
        // Point D
    }

    // Point E
    return z;
}
```

	next < 0	y > z	y == 0
Point A			
Point B			
Point C			
Point D			
Point E			

5. Programming, 15 points.

Write a static method named `numDays` that accepts an integer value in the range 1 through 12 inclusive as a parameter and returns an integer value indicating the number of days in the corresponding month. You can assume the number passed as a parameter is in the range 1-12. The number of days per month should be returned as follows:

Number of Days	Months	Month Number
30	September, April, June, November	9, 4, 6, 11
31	January, March, May, July, August, October, December	1, 3, 5, 7, 8 10, 12
28	February	2

Sample Calls :

`numDays(4)` should return `30`
`numDays(2)` should return `28`
`numDays(1)` should return `31`

6. Programming (15 points)

Write a static method named `threeHeads` that repeatedly flips a coin until three heads *in a row* are seen. You should use the Random class to give an equal chance to a head or a tail appearing. Each time the coin is flipped, what is seen is displayed (H for heads, T for tails). When 3 heads in a row are flipped a congratulatory message is printed. Here are a few possible outputs of calls to `threeHeads`:

```
T T T H T H H H
Three heads in a row!

H H T T T H H T T H T H H T H H H
Three heads in a row!

T T T H H H
Three heads in a row!

T H T H T T T T T H T H H H
Three heads in a row!
```

7. Programming (10 points)

Write a method `numTWords` that takes a String as a parameter and that returns the number of words in the String that *start* with the letter t (upper or lowercase). By definition, words are separated by one or more spaces. For example:

```
numTWords("are   there   lots    of   t   words   here?")
```

should return 2. Notice that words can contain punctuation marks. Any non-empty sequence of non-space characters can be a word. There might be spaces at the beginning or end of the String. For example:

```
numTWords("   how   about  trick-or-treaters,  candy,  and  cats?   ")
```

should return 1. You may not construct any other objects to solve this problem (e.g., you can't use a Scanner). You may assume that the String has no other whitespace characters such as tabs or newline characters. Your method has to pay attention only to spaces to decide how many words there are.