CSE 142, Summer 2013
Midterm Exam, Friday, July 26, 2013

Name: _____

Section: _____ **TA:** _____

Student ID #: _____

- You have **60 minutes** to complete this exam.
- You may receive a deduction if you keep working after the instructor calls for papers.
- This exam is closed-book/notes.
- You may not use any calculators, cell phones, Google Glass, or other computing devices.
- Code will be graded on proper behavior/output and not on style, unless otherwise indicated.
- Do not abbreviate code, such as "ditto" marks or dot-dot-dot ... marks.
  The *only* abbreviations that *are* allowed for this exam are:
  - `S.o.p` for `System.out.print`,
  - `S.o.pln` for `System.out.println`,
  - `S.o.pf` for `System.out.printf`, and
  - A, S, and N for Always, Sometimes, and Never on problem 5.
- You do not need to write `import` statements in your code.
- If you enter the room, you must turn in an exam before leaving the room.
- You must show your Student ID to a TA or instructor for your exam to be accepted.

Good luck!

Score summary: (for grader only)

Problem	Description	Earned	Max
1	Expressions		10
2	Parameter Mystery		12
3	If/Else Simulation		12
4	While Loop Simulation		12
5	Assertions		15
6	Programming		15
7	Programming		15
8	Programming		9
TOTAL	**Total Points**		**100**

1. Expressions (10 pts)

For each expression at left, indicate its value in the right column. List a value of appropriate type and capitalization. e.g., `7` for an `int`, `7.0` for a `double`, `"hello"` for a `String`, `true` or `false` for a `boolean`.

Expression	Value
`12 / 3 + 5 + 3 * -2`	_____
`2 + 2 + "(2 + 2)" + 2 + 2`	_____
`13 / 2 - 38 / 5 / 2.0 + (15 / 10.0)`	_____
`3.0 / 1.5 - 6 / 4 - 10.0 / 2 / 2`	_____
`20 % 6 + 6 % 7 + 1 % 6`	_____

2. Parameter Mystery (12 pts)

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) {
        int a = 2;
        int b = 9;
        int c = 4;
        int three = a;
        int one = b + 1;

        axiom(a, b, c);
        axiom(c, three, 10);
        axiom(b + c, one + three, a + one);
        a++;
        b = 0;
        axiom(three, 2, one);
    }

    public static void axiom(int c, int b, int a) {
        System.out.println(b + " + " + a + " = " + c);
    }
}
```

3. If/Else Simulation (12 points)

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

```
public static void ifElseMystery(int x, int y) {
    if (x == y) {
        x = x + 11;
    } else if (x > 2 * y) {
        x = 0;
    }
    if (x == 0 || y > x) {
        x = x + 2;
        y = y + 2;
    } else {
        y = y - 1;
    }

    System.out.println(x + " " + y);
}
```

Method Call	Output
`ifElseMystery(5, 5);`	_____
`ifElseMystery(18, 4);`	_____
`ifElseMystery(3, 6);`	_____
`ifElseMystery(0, 0);`	_____

4. While Loop Simulation (12 points)

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

```
public static void whileMystery(int x, int y) {
    while (x > 0 && y > 0) {
        y = y - x;
        x--;
        System.out.print(y + " ");
    }

    System.out.println(y);
}
```

Method Call	Output
`whileMystery(5, 7);`	_____
`whileMystery(4, 20);`	_____
`whileMystery(10, 40);`	_____
`whileMystery(5, 15);`	_____

5. Assertions (15 points)

For each of the five points labeled by comments, identify each of the assertions in the table below as either being *always* true, *never* true, or *sometimes* true / sometimes false. (You may abbreviate them as A, N, or S.)

```
public static int stuff(Random r, int m) {
    int c = 0;
    int t = 0;
    int d = r.nextInt(m);

    // Point A
    while (c <= 3) {
        // Point B
        d = r.nextInt(6) + 1;
        if (d <= m) {
            c++;
            // Point C
        } else {
            c = 0;
            // Point D
        }
        t++;
    }

    // Point E
    return t;
}
```

	c > 3	d <= m	c == 0
Point A			
Point B			
Point C			
Point D			
Point E			

6. Programming (15 pts)

Write a static method called `printListMin` that takes a `Random` object and an integer *n* as a parameter and produces two lines of output, where the first line contains *n* randomly generated two-digit numbers, separated by a comma and a space, and the second line contains a phrase indicating the minimum value from the previous line. The method should use the `Random` object to select numbers in the range of 10 to 99 (inclusive) where each number is equally likely to be chosen. For example, given the following lines of code:

```
Random r = new Random();
printListMin(r, 3);
printListMin(r, 5);
```

You would expect output like the following:

```
46, 32, 77
min was 32
88, 47, 54, 62, 19
min was 19
```

You may assume that the value passed to your method is greater than or equal to 1. You are to exactly reproduce the format of these logs, though due to the randomness, your method is unlikely to produce the same values. Write your solution to `printListMin` below.

7. Programming (15 pts)

Write a static method called `randomArt` that takes two integers as parameters, *size* and *frequency*, and prints a *size* by *size* grid where each character in the grid is selected randomly between an asterisk (*) and a plus sign (+), and returns the number of asterisks printed in the artwork. The plus sign should be *frequency* times as likely to be printed as the asterisk. Thus, if *frequency* is 1, the plus sign should be equally likely to be printed as the asterisk, and if *frequency* is 4, the plus sign should be 4 times more likely to be printed than the asterisk. You may assume that the values passed to your method will both be integers greater than or equal to 1. Your method should construct a `Random` object to produce the randomness. Several example calls are shown below (note that, because the method uses randomness, repeated calls will likely not produce the same output).

Call	`randomArt(4, 2);`	`randomArt(7, 5);`	`randomArt(1, 4);`

Example Output	`++++`
	`*+**`
	`+++*`
	`*+**`
		`+++++++`
		`*++++++`
		`+++++++`
		`+*++*+*`
		`*+++*++`
		`++*++*+`
		`++++**`
			`+`

Return value	`6`	`11`	`0`

Write your solution to `randomArt` below.

8. Programming (9 pts)

Write a static method called `countParitySwitches` that takes an integer value as a parameter and returns the number of parity (even or odd) switches between the digits (that is, the number of pairs of consecutive digits where one is even and the other is odd, in either order). For example, a call of `countParitySwitches(36487)` would return 2 because the parity switches between the 3 and the 6 and again between the 8 and the 7. If the value passed contains only one digit, or only even or odd digits, the method should return 0. You may assume that the value passed to the method will be greater than or equal to 1. Below are the results of some sample calls to `countParitySwitches`.

Method Call	Return Value
`countParitySwitches(5);`	0
`countParitySwitches(800);`	0
`countParitySwitches(12345);`	4
`countParitySwitches(14);`	1
`countParitySwitches(75930);`	1

Write your solution to `countParitySwitches` below.