CSE 142, Autumn 2011
Midterm Exam: Friday, November 4, 2011

Name: _____

Section: _____ **TA:** _____

Student ID #: _____

- You have 50 minutes to complete this exam.
  You may receive a deduction if you keep working after the instructor calls for papers.
- This exam is open-book. You may not use any computing devices including calculators.
- 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`, and
  - `S.o.pf` for `System.out.printf`.
- 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		15
3	If/Else Simulation		10
4	While Loop Simulation		10
5	Assertions		15
6	Programming		15
7	Programming		15
8	Programming		10
TOTAL	**Total Points**		**100**

1. Expressions

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
`5 + 2 * 4 / 3 + 5`	_____
`5 + 5 + "23.0" + 5 + 2 * 5`	_____
`!(5 > 2 && -2 > 2) || 5 / 2 == 0`	_____
`15 % 9 % 4 + 4 % 6 % 3`	_____
`12 / 5 / 2.0 + 2 * 4`	_____

2. Parameter Mystery

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 bril = "vorpal";
        String gyre = "jubjub";
        String slithy = "snack";
        String tum = "mut";
        String mut = tum + 1;

        mystery(bril, slithy, gyre);
        mystery(gyre, "gyre", mut);
        mystery(gyre + slithy, bril, tum);
        tum = "tumtum";
        bril = "slithy";
        mystery(tum, gyre, slithy);
   }

   public static void mystery(String gyre, String bril, String slithy) {
        System.out.println("Twas " + bril + " and the " + slithy +
                           " toves did " + gyre);
   }
}
```

3. If/Else Simulation

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) {
   int z = 0;
   if (x >= y) {
        x = x / 2;
        z++;
   }
   if (x > z && y <= z) {
        z++;
   } else if (x > z) {
        y = y - 5;
   }
   System.out.println(x + " " + y + " " + z);
}
```

Method Call	Output
`ifElseMystery(4, 1);`	_____
`ifElseMystery(-10, 100);`	_____
`ifElseMystery(18, 4);`	_____
`ifElseMystery(-12, 5);`	_____

4. While Loop Simulation

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

```
public static void mystery(int x, int y) {
   int z = 1;
   while (x > 0) {
        System.out.print(y + ", ");
        y = y - z;
        z = z + y;
        x--;
   }
   System.out.println(y);
}
```

Method Call	Output
`mystery(2, 3);`	_____
`mystery(3, 5);`	_____
`mystery(4, 7);`	_____

5. Assertions

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.

```
public static int mystery(Scanner console, int f) {
    int num = console.nextInt();
    int h = 0;
    // Point A
    while (f < 5) {
        // Point B
        if (num == 0) {
            h = 0;
            f++;
            // Point C
        } else {
            // Point D
            h++;
        }
        num = console.nextInt();
    }
    // Point E
    return f;
}
```

Fill in each box below with one of ALWAYS, NEVER or SOMETIMES. (You may abbreviate them as A, N, or S.)

	`h == 0`	`f >= 5`	`num == 0`
Point A			
Point B			
Point C			
Point D			
Point E			

6. Programming

Write a static method named `twoConsecutive` that accepts three integers as parameters and returns `true` if there is at least one pair of integers that differ by exactly 1. For example, the integers 3 and 4 differ by 1. The integers 12 and 11 also differ by 1. Your method should return `false` if there are no such consecutive values. The integers could be passed in any order; the two consecutive values could be any of the two values passed in.

Here are some sample calls:

Call	Output
`twoConsecutive(1, 2, 12)`	`true`
`twoConsecutive(1, 12, 2)`	`true`
`twoConsecutive(2, 12, 1)`	`true`
`twoConsecutive(4, 5, 3)`	`true`
`twoConsecutive(2, 4, 6)`	`false`
`twoConsecutive(8, 8, 8)`	`false`

7. Programming

Write a static method named `stitching` that accepts two integer parameters `w` and `h` and that prints a rectangle of dashes and numbers. Each of the `h` lines printed will contain `w` integers separated by dashes. The first number on each line is the line number. For example, the first line's first number is 1 and the sixth line's first number is 6. The lines alternate between starting with a dash and ending with a dash. For example, the first, third and fifth lines start with a dash and the second, fourth and sixth lines start with a number.

You may assume that the value of each parameter is greater than or equal to 1. Your output must exactly match the format shown.

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

Call	`stitching(6, 2);`	`stitching(2, 3);`	`stitching(2, 6);`	`stitching(1, 1);`

Output	`-1-2-3-4-5-6`
	`2-3-4-5-6-7-`
		`-1-2`
		`2-3-`
		`-3-4`
			`-1-2`
			`2-3-`
			`-3-4`
			`4-5-`
			`-5-6`
			`6-7-`
				`-1`

8. Programming

Write a static method named `sameFlip` that accepts a `Random` object as a parameter. Your method should flip a coin until the same result occurs twice in a row. In other words, if a head is flipped followed by another head or if a tail is flipped followed by another tail, your method should end. You should use the `Random` object to give an equal chance to a head or tail appearing. Each time the coin is flipped, print H for heads or T for tails.

For example, if the following variable is initialized:

```
Random r = new Random();
```

Here are some sample calls along with possible output:

Call	Output
`sameFlip(r);`	`HTHH`
`sameFlip(r);`	`HTHTT`
`sameFlip(r);`	`TT`
`sameFlip(r);`	`THTHTHH`