CSE 142, Autumn 2006
Midterm Exam, Monday, November 6, 2006

Name: ____

Section or TA: _____________________

Student ID #: _____________________

Rules:

- You have 50 minutes to complete this exam.
  You may receive a deduction if you keep working after the instructor calls for papers.
- This test is open-book/notes. You must work alone and may not use any computing devices of any kind including programmable calculators. A scientific calculator may be used.
- Cell phones and other electronic devices may NOT be out during the exam for any reason, even if the device includes a calculator program.
- Write legibly and case-sensitively. On programming questions, **do not abbreviate Java code**.
- Your code you write will be graded purely on **external correctness** (proper behavior and output) and not on internal correctness (style) except as indicated in a particular problem's description.
- You do not need to write any `import` statements in your exam code.
- Do not abbreviate any code on your exam, such as writing `S.o.p` for `System.out.println`.

- Please be quiet during the exam. If you have a question or need, please raise your hand.
- Corrections or clarifications to the exam will be written at the front of the room.
- If you violate the University Code of Conduct during this exam, you may receive a 0% for this exam and possibly further punishment.
- When you have finished the exam, please turn in your exam quietly and leave the room.
- If you enter the room, you must turn in an exam paper and will not be permitted to leave the room without doing so.
- You must show your UW Student ID card to a TA or the instructor for your submitted exam to be accepted.

Good luck!

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
`6 * 2 + 8 / 2 * 3`	
`3 * 6 * 2.0 / 5 + 3 / 4`	
`25 % (9 - 5) - 33 % 9 % 4`	
`5 + 2 * 3 + "6" + 15 + 2 + "(1 + 1)"`	
`1000 / 5 / (30 / 7) / 8.0 + 10 / 4.0`	

2. **Parameter Mystery (15 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 pizza = "snow";
        String apple = "skiers";
        String x = "pizza";
        String y = "smoothies";
        String z = "apple";

        smoothies("apple", z, y);
        smoothies(z, x, z);
        smoothies(apple, y, x);
        smoothies(y, "smoothies", pizza);
        String w = z + "-" + x;
        smoothies("pizza", w, x);
    }

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

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 n) {
    int x = 2;
    while (x < n) {
        if (n % x == 0) {
            System.out.print(x + " ");
            n = n / x;
            x = 2;
        } else {
            x++;
        }
    }
    System.out.println(n);
}
```

Method Call	Output
`mystery(1)`	______________________________
`mystery(6)`	______________________________
`mystery(21)`	______________________________
`mystery(24)`	______________________________
`mystery(120)`	______________________________

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 int mystery(int z) {
       int y = 10;
       int x = 0;

       // Point A
       while (z >= y) {
           // Point B
           if (x <= y) {
               // Point C
               x++;
           }

           z--;
           // Point D
       }

       // Point E
       return z;
}
```

Fill in each box of the the table below with one of the following words: ALWAYS, NEVER or SOMETIMES. (You may abbreviate these choices as A, N, and S as long as you write legibly. If we cannot read your answer, it will be marked incorrect.)

	`x <= y`	`y > z`	`x == 11`
Point A			
Point B			
Point C			
Point D			
Point E			

5. **Programming (15 points)**
Write a static method named `rockPaperScissors` that accepts a pair of integers as parameters representing two players' choices in the game of Rock-Paper-Scissors, and returns which player won the game.

Rock-Paper-Scissors is an old game where two people each simultaneously choose one of three possible weapons: a rock, a piece of paper, or a pair of scissors. The paper defeats the rock but loses to the scissors. The rock defeats the scissors but loses to the paper. The scissors defeats the paper but loses to the rock. If the two players choose the same weapon, the game is a tie.

In your code, the two players' weapons are passed as integer parameters. Paper is represented as the integer value 0, rock is represented as 1, and scissors is represented as 2. Your method should examine the two players' weapons and return an integer representing the player that won the game. If the first player wins the game, return 1. If the second player wins the game, return 2. Your method should return 0 in the case of a tie. You may assume that both parameter values passed will be valid (between 0 and 2 inclusive).

Here are some example calls to your method and their expected return results:

Call	Meaning	Value Returned
`rockPaperScissors(0, 1)`	p1 is paper, p2 is rock	`1`
`rockPaperScissors(2, 1)`	p1 is scissors, p2 is rock	`2`
`rockPaperScissors(0, 0)`	p1 is paper, p2 is paper	`0`
`rockPaperScissors(1, 2)`	p1 is rock, p2 is scissors	`1`
`rockPaperScissors(1, 0)`	p1 is rock, p2 is paper	`2`

6. **Programming (15 points)**

Write a static method named `longestWord` that accepts a string as its parameter and returns the length of the longest word in the string. A word is a sequence of one or more non-space characters (any character other than the space character, `' '`).

Here are some example calls to your method and their expected results:

Call	Value returned
`longestWord("to be or not to be")`	`3`
`longestWord("oh hello, how are you?")`	`6`
`longestWord("I am OK")`	`2`
`longestWord("   this   example has     many spaces  ")`	`7`
`longestWord("test")`	`4`
`longestWord("")`	`0`
`longestWord("    ")`	`0`

Note that the string might be empty, might not contain any words, might begin with spaces, and might contain words separated by multiple spaces. You may assume that the string doesn't contain any other whitespace characters such as tabs or newlines. You may not use auxiliary objects such as `Scanner` or `StringTokenizer` to solve this problem.

7. **Programming (15 points)**

Write a static method named `getOddInteger` that accepts as its parameter a `Scanner` for the console and repeatedly prompts the user until an odd integer (an integer not evenly divisible by 2) is entered, then returns this integer. If the user enters an integer that is even, or a token that is not an integer, an error message is displayed and the user is prompted again. You may assume that the user will type a single token as their response each time.

For example, the following log of execution represents the output from one call to your method. (User input is underlined.) After this output, the method returns 17, because this is the first odd integer typed by the user.

```
Type an odd integer: hello
Invalid input, try again.
Type an odd integer: okay
Invalid input, try again.
Type an odd integer: 268
Invalid input, try again.
Type an odd integer: 1.875
Invalid input, try again.
Type an odd integer: -24
Invalid input, try again.
Type an odd integer: 17
```

The following is another example log of execution in which the user types an odd integer on the first try. After this output, the method returns 285.

```
Type an odd integer: 285
```

X. Extra Credit (+1 points)

Draw a picture of somebody in the UW CSE department whom you know. It can be a picture of yourself, one of your classmates, your TA, one of the instructors, etc. Make sure to write their name or description underneath, so it's clear who they are!

This is not an art class, so you will get full credit as long as you don't leave the page blank and your drawing appears to have taken more than a few seconds' work.

Score summary: (for grader use only)

Problem	Description	Earned	Max
1	Expressions		10
2	Parameter Mystery		15
3	While Loop Simulation		15
4	Assertions		15
5	Programming		15
6	Programming		15
7	Programming		15
X	Extra Credit		+1
TOTAL	**Total Points**		**100**