CSE 2221 — Midterm Exam #1
SAMPLE

This is a closed-book, closed-notes, closed-electronic-device, closed-neighbor exam.

In accordance with The Ohio State University Code of Student Conduct, I certify that I have neither received nor given aid on this examination, that I shall not discuss the contents of this examination with anyone in CSE 2221 who has not already taken the exam, and that I have not recorded and taken from the room any questions or answers from this exam.

Name _____ ***Signature*** _____

There are 100 points on the exam plus 5 extra-credit points. Please write your answers on the test sheets, and of course don't forget to write ***and sign*** your name on the top sheet. Consider the space allotted as an indication of the expected length of the answer.

1. (15 points; 3 points each) Multiple choice: circle the best response to each.

   1.1. In design-by-contract, the code responsible for making sure the precondition (requires clause) is true when a method is called is:

   A. the client code that calls the method

   B. the code that implements the method

   C. both the client and the implementation code

   D. neither the client nor the implementation code

   1.2. Consider the following method signature:

```
private static int examScore(int studentNumber) {...}
```

   Here, `studentNumber` is called:

   A. a distinguished variable

   B. an argument (or actual parameter)

   C. a formal parameter

   D. an index

1.3. Suppose the method from question 1.2 is called in the following statement:

```
int k = examScore(42);
```

This call would certainly:

A. be illegal in Java (i.e., it's a compile-time error)

B. cause the program to crash when it is executed (i.e., it's a run-time error)

C. print out the exam score of student #42

D. assign an exam score of 42 to student *k*

E. be legal in Java (though flagged by CheckStyle)

1.4. Suppose you want to set the `double` variable `oneToThree` to a random real number uniformly distributed in the interval [1.0, 3.0). You have made the following declaration:

```
Random r = new Random1L();
```

noting that `r.nextDouble()` returns a random real number uniformly distributed in the interval [0.0, 1.0). Which statement will set `oneToThree` to the desired result?

A.
```
oneToThree = 3.0 * r.nextDouble();
```

B.
```
oneToThree = 1.0 + 2.0 * r.nextDouble();
```

C.
```
oneToThree = 1.0 + r.nextDouble() + r.nextDouble();
```

D.
```
oneToThree = r.nextDouble() + r.nextDouble() +
r.nextDouble();
```

1.5. Suppose you want to display the image file `brutus.jpg` in a browser, along with other content as organized in the HTML file `index.html`. Both these files are in the same folder/directory. Which HTML code fragment in `index.html` will produce the desired result? (Note that a well-formed HTML document such as the one you were to create for project #1 is an XML document.)

A.
```
<html>brutus.jpg</html>
```

B.
```
<img src="brutus.jpg" />
```

C.
```
<p type="image">brutus.jpg</p>
```

D.
```
<a href="brutus.jpg" />
```

2. (3 points) Short answer: give a clear and concise response.

2.1. (2 points) When CheckStyle warns you about a "magic number" in your code, what does this mean? Explain briefly why this warning is something you, as a software professional, should pay attention to and consider fixing (even if you decide not to change your code to remove the warning for a mere class assignment).

2.2. (1 point) Show the value of *x* after this statement: `double x = 1/2;`

3. (17 points) Complete the tracing table below.

Code	State (Variable Values)
`int i = 1;`	
	`i =`
`int j = 2;`	
	`i =` `j =`
`while (i < j) {`	
	`i =` `j =`
`    i = i * i + 1;`	
	`i =` `j =`
`    j = j + 2;`	
	`i =` `j =`
`}`	
	`i =` `j =`

4. (20 points)

4.1. (15 points) Write the body of this method.

```
/**
 * Returns the smallest value in a.
 *
 * @param a
 *            an array of ints
 * @return smallest value in the array a
 * @requires
 * a.length >= 1
 * @ensures
 * minimum = [the smallest value in a]
*/
private static int minimum(int[] a) {









}
```

4.2. (5 points) Clearly and concisely explain why the precondition is important here.

5. (10 points) Trace the effect of the line of code below that calls `squareAndReset` by tracing the execution of the method body of `squareAndReset` for this call.

Tracing table for the call:

Code	State (Variable Values)
	`num = 13.0`
`double sq = squareAndReset(num);`	
	`num =` `sq =`

Tracing table for `squareAndReset`:

Code	State (Variable Values)
`private static double squareAndReset (double x) {`	
	`x =`
`    double square = x * x;`	
	`x =` `square =`
`    x = 0.0;`	
	`x =` `square =`
`    return square;` `}`	

6. (20 points) Complete the body of `main` so it prompts the user for a password candidate until either the password candidate is the empty string, or the password candidate is "acceptable" according to `isOK`. (In our formal specifications, for a mathematical string such as `pwd`, `|pwd|` means the length of `pwd`.)

```
/**
 * Checks whether the given string satisfies the criteria for
 * an acceptable password.
 *
 * @param pwd
 *            the string to check for acceptability
 * @requires
 * |pwd| > 0
 * @ensures
 * isOK = [pwd satisfies the criteria for an acceptable password]
 */
private static boolean isOK(String pwd) {...}

/**
 * Main method.
 *
 * @param args
 *            the command line arguments
 */
public static void main(String[] args) {
    SimpleReader in = new SimpleReader1L();
    SimpleWriter out = new SimpleWriter1L();









    in.close();
    out.close();
}
```

7. (9 points) The following is a valid XML document.

```
<?xml version="1.0"?>
<buckeyes>
  <location stadium="home" />
  <game opp="UAB" date="22 Sep 2012">
    <forecast>Sunny with 90% probability of football</forecast>
  </game>
</buckeyes>
```

7.1. (8 points) Draw the `XMLTree` associated with this document. (Don't worry about drawing the "tag" icon; just include the enclosing brackets `<...>` for each label that is a tag.)

7.2. (1 point) How many children does the root node of this `XMLTree` have?

8. (6 points) The following is not a valid XML document. Circle the locations of three non-overlapping errors that make it invalid, briefly explaining each error in the space below.

```
<?xml version="1.0"?>
<pony>
   <unicorn mark="three lozenges">Rarity
     <color>cyan</unicorn>
   </color>
   <unicorn mark="six-pointed star">Twilight Sparkle</unicorn>
   <unicorn mark="two dolphins" version="G4" />
</pony>
<dragon>
   <youth>Spike
</dragon>
```

9. (5 points ***extra credit***) Which area is larger: the shaded area in the figure on the left (a circle inscribed in a square) or the shaded area in the figure on the right (a quarter-circle inscribed in the same size square)? Justify your answer in clear and concise terms.