CPSC 233 Winter 2013
Assignment Guidelines

In this course, you must submit your own work for all assignments. Please refer to the university calendar and `http://pages.cpsc.ucalgary.ca/~tamj/233/assignments/misconduct.html` for more information and policies on academic misconduct. When working on your assignments, please adhere strictly to the following submission and programming guidelines unless otherwise noted.

Submission Guidelines

For fairness and consistency, all programming questions will be marked on the computers in the computer lab (1st floor MS). You may work on the programming questions on your own computer, but please make sure to test your program under your CPSC account before submitting the assignments. If a program does not work on the lab computers, it is considered incorrect and you will not get full marks. It is important that you do not customize your CPSC account since the TAs will test your program with the default account configuration.

The submit system allows you to submit any number of files, view the files that you have submitted, and replace files that you have submitted until the due date has passed. To access the system, you must log on to one of the CPSC servers or directly work on a CPSC lab computer. Late assignments will not be accepted except in cases of extreme personal emergencies. Penalty may be applied to late submissions.

It's your responsibility to check that the file you submitted is what you intended to be marked based on. You may list the files that you have submitted using the `showstuff` command. Please visit `http://pages.cpsc.ucalgary.ca/~tamj/submit.html` for instructions on using the submit system.

Programming Guidelines

Your program should be formatted for readability and ease of understanding. This includes consistent indenting and placement of braces. You should also demonstrate the approach use of whitespace to group related code blocks (e.g. methods, loop and branches) and to separate unrelated blocks.

- All programming assignment questions should be submitted as *.java files.
- Use variable names that are meaningful. For example, a variable that is used to count the number of characters should be called `numChars` rather than `x` or `num`. Variable names for loop counters can be `i`, `j`, and the like.
- Avoid the use of literal constants. Use named constants instead. For example, in Assignment 1, a constant variable should be use to store the maximum number of allowed guesses.
- In-line documentation is very important when writing programs of large scale. Place descriptive comments prior to major blocks of code and functions, especially for code segments that is subject to misinterpretation. The better your program is written, the better your code is indented, the better variable names are, the less need there will be for comments.
- A comment as follows must appear at the beginning of each and every program you write:

```
/**********************************************************
 * Last Name:  <your last name>
 * First Name:  <your first name>
 * Student ID: <your student ID number>
 * Course: CPSC 233
 * Tutorial Section: <your tutorial sction>
 * Assignment: <assignment number>
 *
 * <Brief description of your program and main features>
**********************************************************/
```

- Unless otherwise told, your program should not have any `static` methods, except for the `main()` method.
- Except for Assignment 1, programs consisting of a single class with a single method is not acceptable. The classes in your program should be logically designed. For example, it is not acceptable to having all or most of the program functionalities in 'super' classes.
- Output should be easy to understand, and match the format specified in the assignment handout.
- Indent your code to make your code more readable, and be consistent. For indentation, it is recommended to use soft tabs (four white spaces) instead of a real tab, as tabs are defined different across operating systems.

- Your program should print an identification banner prior to any output being produced. Once we learn how create methods in Java classes, you should create a method for printing the banner.

```
System.out.println(‘Last Name:  <your last name>’)
System.out.println(‘First Name:  <your first name>’)
System.out.println(‘Student ID: <your student ID number>’)
System.out.println(‘Course: CPSC 233’)
System.out.println(‘Tutorial Section: <your tutorial sction>’)
System.out.println(‘Assignment: <assignment number>’)
```