### CS 203 HW#5:

**Due:** **Wednesday, February 20th at 11:00pm**

This assignment is intended to give you practice with arrays in Java. In this assignment, you will implement a Hangman game. The goal of Hangman game is for the player to determine a secret word by guessing its letters. Initially, the player only knows how many letters are in a word. For example, a five-letter word is displayed as “\_ \_ \_ \_ \_”, where each underscore (‘\_’) represents a letter in the word. The player guesses one letter at a time, and if that letter is in the secret word, all occurrences of that letter are revealed. For example, if the player guesses ‘a’, then assume the following is revealed: “\_ a \_ \_ a”. If the guessed letter is not in the secret word, that letter is added to a list of incorrect guesses, and a part of the Hangman picture is drawn. If a player can determine the word before making seven incorrect guesses, the player wins. To play hangman game go to this link <http://www.playhangman.com/>

**Program Functionality**

The Hangman class implements this game in the main method, which uses two additional helper methods.

1. Review the code provided. Familiarize yourself with the code sections, the flow of execution, and the constants and variables used. The following is a summary of the program’s main loop that repeats so long as the game hasn’t been won and the player hasn’t used all their guesses:
   1. Ask the user if they would like to play the guessing game. If the user's answer begins with a 'y', then continue otherwise quit.
   2. Call the printHangman method (which is DONE for you) to display the current hangman picture.
   3. Print the secret word so far showing the letters that the user has guessed and the other letters as underscores.
   4. Print the incorrect letters so far.
   5. Ask the user to guess a letter.
   6. Use the Scanner object to read in the letter and then handle the guessed letter to check whether it is in the secret word or not.
2. The helper method printHangman, prints the hangman picture using characters. Again, drawing the hangman is already implemented for you.

**Methods**

The starter code contains declarations of all the methods you'll need to write this program. Some of these methods have already been implemented. Others are empty stubs. To complete this assignment, you must implement the stubbed-out helper methods defined in the starter code. You are welcome to add additional methods as you see fit. When you are finished, all of the helper methods should be called at least once somewhere in the code. Read the comments on the methods to learn what they should do.

**Managing the words**

Your program is required to keep track of a guessed characters so far. The starter code has declared some arrays for you. The arrays are as follows:

* String[] secretWordsList - This array contains the secret words that you can choose from for the game.
* char[] correctLetters – This array contains the correct letters that the user guessed so far
* char[] incorrectLetters – This array contains the incorrect letters that the user guessed so far.

**Grading**

The starter code has some comments with TODO tags. You need to complete all the functionality that is tagged as TODO. The grading will be based on the correctness of the following:

* [15%] complete the initialization in the main method
* [20%] complete while loop in the main method
* [25%] implement printCorrectGuessedLetters() method
* [10%] implement printIncorrectGuessedLetters () method
* [10%] implement gameWon() method
* [20%] code quality (see below)

**Additional Enrichment (optional)**

If you have programmed before, or if you like a challenge and have the time, feel free to get a little fancier. If you do, be sure to document the additional features in your code. Here are some ideas:

* Modify the program to accept uppercase and lowercase letters
* Modify the program to display an error if the user enters in a character that is not a letter.
* Modify the program to not allow the user to guess the same letter more than once.
* Modify the program so that the user may play the game until they desire to quit.
* Display a count of the wins and losses.

**Code Quality**

A good computer program not only performs correctly, it also is easy to read and understand:

* A comment at the top of the program includes the name of the program, a brief statement of its purpose, your name, and the date that you finished the program.
* The comment header should also include a list of known bugs or deficiencies, or a statement that the program has no known deficiencies.
* Variables have names that indicate the meaning of the values they hold.
* Code is indented consistently to show the program’s structure.
* The body of if and else clauses are enclosed in braces and indented consistently, even if they consist of only a single line of code.
* Opening braces are placed consistently, either at the end of a statement or on a line by itself directly after it.
* Within the code, major tasks are separated by a blank line and prefaced by one or more single-line comments identifying the task.
* Methods are separated by blank lines and prefaced by a multi-line comment describing what they do and what their parameters mean. (See starter code for examples.)
* Very long statements (such as long print statements or complex boolean expressions) are broken across lines and indented to show their structure.
* Now that you are familiar with loops and methods, your code should not contain redundant or repeated sections.
* Your program does not contain extraneous or commented out code. It does not contain out-of-date or irrelevant comments.
* When a constant value is used repeatedly in your program, declare a final variable with a descriptive name to contain that value. Use it instead of a literal.

**Logistics**

To begin the assignment, download the starter program (Hangman.java) from the course website. Create a new BlueJ project. Then drag and drop Hangman.java into the project. **Note, the current code has some errors, so it will not compile. Make sure to fix these errors before continuing to develop.**

**Turning in this Assignment**

You are responsible for turning in your homework assignments properly.

* Be sure your name is in the comment header at the top of your .java file.
* If you did any of the Additional Enrichment (above), carefully document this functionality in the comment header at the top of your .java file.
* Zip up your entire BlueJ project folder and submit the zip file via Moodle.