CMPSC 111
Introduction to Computer Science I
Fall 2016

Practical 6
October 21, 2016
Due in Bitbucket by midnight, October 24, 2016 (extra time provided)
"Checkmark" grade

Summary

In this practical you will write a Java program that will allow the user to guess a number. To complete this task you will use `while` loops and `if/else` statements. Then, using the "`git add`", "`git commit`", and "`git push`" commands you will upload your modified source code files and the output you obtain from running your program to your Git repository hosted by Bitbucket.

Review the Textbook

You should refer to Sections 5.1–5.4 in your textbook to learn more about `if/else` statements and `while` loops. You may also refer to the section in this document called "A `while` Loop Overview" for an example of using a `while` loop in a Java program. To review the details about random number generation and the `java.util.Random` class, you can study Section 3.4. Please see the course instructor or a teaching assistant if you have questions about the concepts underlying iteration or recursion or if you are unsure of how to realize these fundamental ideas in a Java program.

Implementing a Guessing Game

Write a Java program that will play a guessing game with the user of the program. The user must try to guess a number between 1 and 100. If the user's guess is wrong, print out a helpful hint, such as "that's too high" or "that's too low". Keep repeating this until the user guesses correctly.

Your Java program should complete the following tasks:

- Print your name, practical number, and the date (so, remember to use "`new Date()`").
- Declare and initialize a variable of type `int` to be equal to some randomly generated number in the range between 1 and 100; this is the correct answer that the user will try to guess.
- **Repeatedly** ask the user to enter a number until the user is able to guess the correct answer.
- Print out an appropriate statement if the user's input is equal to the correct answer.
- When the user guesses incorrectly, print out "Too low!" or another appropriate statement if the number entered by the user is less than the correct answer.
- When the user guesses incorrectly, print out "Too high!" or another appropriate statement if the number entered by the user is greater than the correct answer.
- Print out how many tries it took the user to guess the correct answer (so, to complete this task, you will need to declare and increment an additional `int` variable).

Completing the Practical Assignment

Create one Java file, called `GuessMyNumber.java`, for this assignment. This class will contain the `main` method where you will implement a solution to the problem outlined on the previous page.

To finish this assignment and earn a "checkmark", you should submit the `GuessMyNumber.java` file in your Bitbucket repository by using the appropriate `git` commands. You also need to submit an output file, called `output`, with at least three runs of your program. Remember, to best preserve the output's format, you should create and edit the `output` file using the `gvim` text editor.

A `while` Loop Overview

The general form of a `while` loop is:

```
while ( condition )
{
        ... one or more Java statements ...
}
```

As long as the *condition* is true, the body of the loop will execute, and the loop will keep repeating. The condition is tested at the "top" of the loop. Here is an example that shows how to keep generating random throws of a pair of dice until a "double" (both dice are the same) is rolled:

```
int d1 = -1, d2 = -2; // two dice (initialized to different values)
int numTries = 0;
while (d1 != d2) // keep generating random rolls until two are equal
{
    d1 = rand.nextInt(6)+1;
    d2 = rand.nextInt(6)+1;
    numTries++;
}
System.out.println("After " + numTries + " rolls of the dice, two were equal");
```

General Guidelines for Practical Sessions

- **Submit *Something*.** Your grade for this assignment is a "checkmark" indicating whether you did or did not complete the work and submit something to your Bitbucket repository using the "`git add`", "`git commit`", and "`git push`" commands.
- **Practice Key Laboratory Skills.** As you are completing this assignment, practice using the `gvim` text editor and the Ubuntu terminal until you can easily use their most important features. Additionally, ask a teaching assistant or the course instructor to teach you some of the advanced features of `gvim` and the terminal, thereby helping you to work more effectively.
- **Update Your Repository Often!** You should `add`, `commit`, and `push` your updated files each time you work on them, always including descriptive messages about each code change.
- **Review the Honor Code Policy on the Syllabus.** Remember that while you may discuss your work with other students in the course, code that is nearly identical to, or merely variations on, the work of others will be taken as evidence of violating the Honor Code.