**Object Oriented Programming I**

**Number Guess – Loop Programming Assignment**

In this assignment you are going to program a number guessing game in which you have 10 chances to guess a number between 1 and 100. A sample run is show below:

**Sample run**: Suppose the random number is 68.

*You have 10 tries to guess a number between 1 and 100*

*Guess number 1: 50*

*Your guess is too low.You have 9 tries left.*

*Guess number 2: 75*

*Your guess is too high.You have 8 tries left.*

*Guess number 3: 65*

*Your guess is too low.You have 7 tries left.*

*Guess number 4: 70*

*Your guess is too high.You have 6 tries left.*

*Guess number 5: 68*

*Congratulations! You correctly guessed the number is 5 tries.*

*Good score!*

In the program you will need a random number between 1 and 100. Review the section on generating random numbers in your text. Use *printf* rather than *println* statements for most of your output statements.

Your program should print a message informing the user of the quality of their score. To score the user’s performance, use the following:

|  |  |
| --- | --- |
| Number of Tries | Message |
| Between 1 and 3 tries | Excellent score |
| Between 4 and 6 tries | Good score |
| Between 7 and 9 tries | Average score |
| 10 tries | Lousy score |

Just because a block of code has a definite number of iterations doesn’t mean that you need to use a *for* loop. The fact that you have ten guesses may lead you to think that you need ten iterations. But you might guess the number your first try. So, you can’t tell how many iterations you are going to need when you start the game. Therefore, this program can be more naturally written with a *while or do - while* loop.

In this program you are going to use a Boolean variable to control the main loop. You should review the section in your text and the sample programs that discuss using Boolean variables for loop control.

**To practice different types of loops, we are imposing a structural constraint on this problem. This program requires a nested loop structure. The inner loop is a do-while data validation loop that checks to make sure the user enters valid value between 1 and 100. The outer (main) loop is a *while (! done)* loop that controls the game play using the Boolean variable *done*. This means you are not free to write this program anyway you please. Programs that do not use this structure will not meet specifications and will have to be redone.**

In other words, your main method should have the following structure (pseudocode):

***while (! done) /****/ outer loop*

*{*

***do***

*{*

*get and validate a guess*

*}* ***while (not valid)*** *// inner validation loop*

*Logic to process guess*

*}*

Your program should only accept integers between 1 and 100. Review the section on do-loops in your text for an example of how to do this. If the user tries to guess a number outside of this range, you should print an error message and prompt the user for a new guess. Notice that an out-of-range guess does not count against the number of guesses.

Sample data validation:

*You have 10 tries to guess a number between 1 and 100*

*Guess number 1: -10*

*Guesses should be between 1 and 100*

*Guess number 1: 110*

*Guesses should be between 1 and 100*

*Guess number 1: 50*

*Your guess is too high. You have 9 tries left.*

*Guess number 2:*

If the player loses the game, you should print a message and display the number. Notice that your *printf* statement for the last try prints the singular “try”, not the plural “tries”.

*Guess number 9: 1*

*Your guess is too low. You have 1 try left.*

*Guess number 10: 12*

*Sorry, you did not guess the number in 10 tries.*

*The number was 77.*