CS-3502 Programming Assignment, Winter AY-18
(15% of total course grade)

You're to develop a networked computer game called "Guess that Number!" You're expected to use Python; but other languages are permitted with the consent of the instructor.

You're to develop both the server and client programs. The game host starts the game by starting the game server on a specific port (you may choose to fix that port number or make it customizable on the command line). However, the server's IP address must be a command line option or input via a GUI for the client program. Once started the server picks an integer randomly from 1 to 100 and waits for a fixed amount of time in seconds (variable name: **game_time**) for the players to join the game. The **game_time** variable must be customizable from the command line for launching the server. The server should not start counting down the time until at least two players have joined the game.

A player joins the game by starting the client program. Once joined, the client prompts the player to input an integer from 1 to 100. (Advanced programmers may want to allow the player to change his/her guess multiple times; just for bragging rights, no bonus points ☺)

At the end of the **game_time** period, the server compares all the guesses from the players who have joined and discloses the outcome to all players using messages such as "The number is <> and your guess is <>. You win with the closest guess / Better luck next time". The server should start another game immediately after that.

Three-step approach:

A divide-and-conquer approach is suggested as follows:

Step 1: develop a system for one player only – the server generates a random number and simply echoes the player's guess and reveals its number to the player at end of **game_time**.

Step 2: Enhance the server code from step 1 to support multiple players using a separate thread for each player.

Step 3: Enhance the server code further to support game logic and starting of new game.

We will conduct labs following this approach. We will use Python examples exclusively in the labs.

Grading:

- You need to provide via email the source code and instructions for testing your game
- You need to turn in your intermediate code in the following schedule:
    - Feb 2: Complete the following warm-up tasks (25% of assignment grade)

Modeling after the code examples presented in class, you need to develop a client server program using both UDP and TCP sockets. You may hard code the server address and port number. The client and server programs enact one of your favorite dialogs between two characters in a movie/TV-program/book. Both sides need to speak and they need to exchange a minimum of one and half rounds of words (i.e. A-> B, B->A, and A-B again). Each side prints every message received from the other side.

- Feb 12: Compete Step 1 tasks (25%)
- Feb 26: Complete Step 2 tasks (30%)
- Mar 5: Complete entire assignment (20%)