CSCI 1133, Spring 2015 Assigned: Tuesday April 28, 2015
Programming Assignment 12 Due: 11:55pm Thursday May 7, 2015

This is ***not*** a collaborative assignment; you must design, implement and test the solution(s) on your own. You may not consult or work with anyone other than the course instructor or TAs. In addition, you may not include solutions or portions of solutions obtained from any source other than those provided in class. Obtaining or *providing* solutions to any homework problems for this class is considered academic misconduct. If you are not sure what this means, consult the class syllabus or discuss it with the course instructor.

This assignment requires writing a single Python module that must be submitted online *prior* to the due date/time. Late submissions will not be accepted. Name your module: `hw12.py` and submit it using the appropriate homework submission link on the Moodle website.

The total point value for programming assignments will be awarded for solutions that are *complete*, *correct*, and *well constructed*. A *"well constructed"* program entails good design, appropriate comments and general readability (descriptive names for variables and procedures, appropriate use of blank space, etc.). The following will result in a score reduction equal to a percentage of the total possible points:

- Incorrectly named/submitted source file (10%)
- Constraints not followed (40%)
- Failure to execute due to syntax errors (30%)

Note that your work will be graded using, and must function correctly with, the current version of Python 3 on CSE Labs UNIX machines. If you complete your programming assignment using a different system, it is your responsibility to ensure your programs work on CSELabs machines *prior* to submitting them.

A. (60 *points*) **Snake Game**

This is a complete programming project in which you will implement an animated Python GUI program to play the classic video game *Snake*. *Snake* is based on one of the first arcade games named *Blockade* from the mid 1970's and became popular when it was re-introduced as one of the early smart phone game apps. To see the game in operation, watch the following YouTube video (warning: turn the audio off, it's obnoxious!):

http://www.youtube.com/watch?v=cpBqFAvLPRs

The game involves controlling the direction of a moving "snake" within a bordered window using the up, down, left and right arrow keys (or the w, s, a and d keys respectively). The "snake" consists of a series of rectangular segments. Additionally, a "food pellet" (represented by a filled circle) is placed randomly within the window. The objective is to maneuver the head of the moving snake so that it collides with a food pellet without touching a window border or any other snake segment. When a food pellet is "consumed", an additional segment is added to the tail of the snake, the speed of the snake is increased and another food pellet is placed randomly in the window. The game begins with a single snake segment (rectangle) and ends when the head of the snake collides with a window border or any other segment of the snake. The objective of the game is to create the longest snake possible without running into the borders or itself.

Write a GUI based Python program using the `tkinter` library that will play the game of *Snake*. Your program needs to do the following:

- Implement the game as a user-defined class named `SnakeGUI` and submit your class definition source code as your solution (do not include any test code).
- Allow either up, down, left, right arrow keys or 'w', 's', 'a' or 'd' keys to control the snake movement.
- Include a selectable button that will pause/restart the game at any point. The label text of the button should be "Start" when the game is not running and "Pause" while it is running.
- When the snake collides with a border or another snake segment, stop the game and display a message indicating "game over" with the total score (the length of the snake). At this point, you should make a provision for the player to restart a new game.

- Start with an animation delay time of 500 milliseconds and reduce it by 10 milliseconds as each food pellet is "consumed".
- Each snake segment should be a square measuring 20 pixels on a side and the game plane should consist of 30 x 30 segments.
- The initial game configuration should consist of a stationary single segment in the middle of the game plane and a single (randomly placed) "food pellet". The game state should be "paused" with the button label set to "Start". Begin moving the (single segment) snake when the button is "clicked".
- Optionally, you can also use the space-bar on the keyboard *in addition* to the button press to start/stop the game.

Helpful Hints:

- Use a canvas widget for drawing food pellets and snake segments and use the canvas .after() method to time the animation.
- Note that you do not have to redraw the entire snake at each time step (although you are free to do so if you wish). You can simply draw the new "head" segment and then erase the old "tail" segment. When a food pellet is "consumed", you increase the snake length by drawing the new head, without deleting the tail.
- Bind the keypress event to the top-level window to avoid keyboard focus issues.
- Keep track of the object identifiers as they are drawn and use them to delete them when needed.