CISC106 Summer 2012 Project

Your job is to implement *Conway's Game of Life (GoL)*. GoL is a grid system that shows that a set of simple rules can cause some complex interactions. The basic program is already designed, and the functions enumerated, so that you will be given certain functions to write. You will also be writing unit tests[1].

Readings

- Review your class notes about lists, dictionaries, pretty much everything we've done so far, and (if we go over it in class) the project.
- `http://en.wikipedia.org/wiki/Conway's_Game_of_Life`

Jul 23

Team rosters due. Team must submit declaration of team name and member names via a single email (see email instructions below) to professor by 11:59 p.m.

Example:

```
Subject: 106 Project

Team members:
Xenon Raqzx, xraqz
Zorpon Minzq, zminzq
```

Aug 10

Complete Project and testing due before midnight.

Recommended timeline:[2]

1. By **July 20**, have the tests for `blank_board` thought out and written up. You'll already know everything you need to know to implement this function after the week of July 9.
2. By **July 23**, have `blank_board` implemented (so that all tests pass)
3. By **July 27**, have `get_neighbors`, `seed_game`, and `create_game` (with just a hardcoded configuration inside the function - I'd wait until last to do the file processing stuff.) tested and working. This is the suggested order to do these:
   (a) `get_neighbors`
   (b) `seed_game`
   (c) `create_game`
4. By **Aug 1**, have `live_count`, `live_neighbors`, and `tick` tested and working. Here is the suggested order on these:
   (a) `live_count`

[1] And of course, as has been the theme of this course, you'll thoughtfully write the tests first and use them to test your code as you develop it!

[2] Note that you can work on whatever timeline you like so long as the entire project is handed in by the due date. However, it is strontly recommended that you follow this timeline to make things as easy as possible on yourself.

(b) `live_neighbors`, since it uses both `live_count` and `get_neighbors`

(c) `tick`, since it pretty much relies on all the code you've written before it.

5. By **Aug 6**, have the file I/O code finished.

   (a) First do the file input code in `create_game`. You should be able to make a file which represents the configuration you hard coded in your first version of `create_game`, and then modify your test slightly to use this file to test the modified version.

   (b) Test and implement `write_cell` since it's needed by . . .

   (c) . . . `write_state`

6. At this point, You should have a mostly working implementation of Conway's Game of Life. All you need to do is implement the drawing code and put everything together.

How will we be graded?

Each team will receive a team grade for the project. The first ninety percent of the grade will be based upon thorough testing and correct completion of the existing functions making up the project. The last ten percent will be based on meeting the team roster deadline.

Email

Any email about the project to myself or Mustafa must include [106 PROJECT] in the subject line, including the square brackets. Emails without this run the risk of being lost or accidentally overlooked[3].

What must be done before June 25?

Nothing, that was weeks ago! See deadlines and recommended timeline for this project above.

Teams

Teams will consist of 2 or 3 people.

NOTE: *You* form teams, but only the professor can break a team apart. Individuals are *not* allowed to leave a team without the professor's permission. Teams are *not* empowered to "vote people off". Form carefully - breaking up is hard, and nobody wins.

Communication

Work on the project within your team only. Seek help and explanations from your team members, not from members of other teams. In particular, DO NOT communicate with another team or outside person about how they wrote a particular function or test. Feel free to send your code or thoughts to other members of your team, but not outside the team. Violations of this policy will be considered academic dishonesty (see the class web page).

All team-related emails[4] to another member of the team must be cc'ed to every team member. This means that everyone will need to be especially careful when choosing their words to comment on someone else's code, timeliness, new shoes, etc.

[3] Adding this information to your subject line allows us to categorize emails automatically. Help us help you!

[4] This rule applies to any written words, including text messaging, paper, etc.