Project 1 Options

October 6th, 2015 — Due: November 3rd, 2015

Choose one of the four project options below. You are encouraged to add additional functionality beyond than what is asked for. If you do only the bare minimum, you must complete everything precisely to get full credit. If you add additional functionality beyond the bare minimum, a few problems or missing parts may be overlooked and you may still receive full credit. Back up your code online frequently. "My code was accidentally deleted"/"My computer broke" will not be a valid excuse for late turn-in. Do not copy code from elsewhere (though an individual line or two is ok).

Text-based Adventure Game

Create a text-based adventure game that a user will be able to play through.

Minimum Requirements

- Create a class for the player and a class for an enemy to hold information like the number of hit-points remaining.
- Include at least 2 save points and allow the player to leave the game and load from them again later (write and read from an external file to do this).
- Include multiple turn-based battle sequences where the player chooses an option during their turn in the battle.
- Allow the player to choose from at least two different paths in the game that will change at least part of the storyline.
- Include a player inventory in which the player can acquire at least 1 item which at some point the game will check whether or not they have the item in their inventory. This inventory should be part of the player class.

Graph Search

Create a class that can preform basic graph theory functionality. You only need to consider simple, undirected graphs. In other words, there's no need to worry about directed edges, loops, or parallel edges.

Minimum Requirements

- Create a class for graphs.
- Create a function that will make a new graph from a file containing a list of edges. A single edge will be given in the file by the two vertices it connects (i.e. "5,4" says there is an edge that connects vertex 5 and 4). The vertex information should be stored within the graph class object.
- Implement a function, that starting from a specific vertex will preform breadth first search on the graph. Each vertex's parent vertex in this search should be the result.
- Implement a function, that starting from a specific vertex will preform depth first search on the graph. Each vertex's parent vertex in this search should be the result.

”Virus”

Create a program that wreaks havoc and steals information.
Important: Be cautious when working on this project and back up your code online. Do not have the code delete (or overwrite) files until you know the program works correctly. Accidentally having your program delete itself or breaking your computer will not be accepted as an excuse for not turning in the project on time.

Minimum Requirements

- Have the program open every file in a directory and all sub directories.
- Email the content from each of these files to an email account (you can assume all files are basic text files that are not extremely long). Emails can be sent from Python as described here: `http://en.wikibooks.org/wiki/Python_Programming/Email`
- Email the file structure of the these files to the address as well. This structure just needs to make it clear what files were in what folders (and what parent folders those folders were in). The way you present this information is up to you. (**Note:** if you use this approach with Gmail you'll have to turn on the ”less secure apps” option in Gmail, so I'd recommend making an extra Gmail account specifically for this project).
- Replace the content of each file you read with a copy of your programs code and change the extension to ”.py”.

Simplified Checkers

Create a simplified checkers game.

Minimum Requirements

- Allow two players to take turns inputting commands.
- Display the current board state on every turn (multiline ASCII string is an easy way to do this). This state information should be stored in a Board class object.
- Check that a move is legal.
- You only need to handle forward moves and single jumps. There is no need for the forced jump rule.
- If no pieces can move, count and display the number of pieces each player has left.