EECS 391: Introduction to AI (Spring 2020) Programming Exercise 1

General Instructions:

Please comment your code extensively so we can understand it, write efficient code using good data structures and sensible variable names. We will use the git logs to ensure that each person is contributing equally to all exercises. Therefore, any code written by you must be clearly logged under your own name/network ID by git. You will not receive credit for code committed by anyone other than yourself even if you wrote it, or if we cannot make out who committed the code. **There will be NO exceptions to this rule.**

The redmine system used by csevcs has issues with some Unicode characters. Please ensure your git names use only standard ASCII characters.

Your commits are due on csevcs.case.edu by 11:59pm on the due date specified after the question. You will receive a 10% bonus for any solution turned in a week or more in advance of the due date. You must notify us of such a commit by email. You can use one late day each week (up to Saturday 11:59pm) with a penalty of 20%. Submissions after Saturday 11:59pm for any week will not be graded. Other bonus points may be awarded for exceptionally well-written and commented, easy to read, clean and efficient code.

These programming exercises will use SEPIA ("Strategy Engine for Programming Intelligent Agents"), a game environment written by other CWRU students tailored to writing AI players. SEPIA is a real-time strategy (RTS) game (though we will not use the real-time aspects in these exercises). RTS games generally have "economic" and "battle" components. In the economic game, the goal is to collect different types of resources in the map. Typical resources are "Gold" and "Wood." Resources are collected using units called "Peasants." Having resources allows the player to build other buildings (Peasants can be used to build things) and produce more units. Some of these units are battle units that can be used to fight the opponent. Games generally end when one player has no more units left; however, in SEPIA, a variety of victory conditions can be declared through XML configuration files. For example we can declare a victory condition to be when a certain amount of Gold and Wood have been collected, some number of units of a certain type built, etc.

You need Java 1.8 to run SEPIA. Note that, although the components of SEPIA you will use have been fairly well tested by now, there is still a possibility of bugs remaining. If you encounter behavior that seems strange, please let me or the TAs know.

Programming 1: Familiarization with SEPIA (Due 2/7 (50 points))

This assignment must be done by each person individually.

Read the documentation at http://engr.case.edu/ray_soumya/sepia/html/ and go through the exercises of building and compiling a simple resource collection and combat agent. Using the same framework, write simple extensions to the resource collection and combat agent. For example, you might try building a Farm (500 gold, 250 wood), a Barracks (700 gold, 400 wood) and a Footman or two (600 gold) in the resource scenario (25 points), or coming up with a smarter attack strategy in the combat scenario (25 points).

If you see errors such as "No class found" or "Unable to instantiate" when trying to run SEPIA, check that (i) you are running Java 1.8, (ii) your classpath is correctly specified, (iii) code for your agents and the enemy agents provided are in the correct locations as expected by their packages and (iv) you are using the correct XML config file (the maps are also XML, so check that you did not specify the map as the config).

Your git repository is located at https://csevcs.case.edu/git/2020_spring_391_N, where N is your Canvas group number. Each person should create a separate subdirectory "P1agents_abc123" in your git repository, where abc123 is your CWRU ID. Place your agent code in it and push to csevcs. Include a short README file containing a brief description of what your modified agents do, your experience with the documentation, and anything you found confusing. Do **NOT** commit any files other than the java files containing your agent and the README.