SYSC 3110 Project – Monopoly!

The goal of this team project is to reproduce a simplified version of the classic board game Monopoly. If you don't know the game, you can play some knock-off versions online. The rules can also be found on Wikipedia.

We will simplify the game by eliminating the Chance/Community chest cards and by not allowing mortgages nor sales and trades between players.

The cost of rent, houses and hotels will be simplified to represent fixed percentages of the initial cost of the property (the amount you choose for these percentages is up to you).

The project is divided into 4 iterations, each ending with a milestone corresponding to deliverables that will be graded. You will be able to use the TA feedback from iteration i for iteration i+1.

Milestone 1: A text-based playable version of the game, i.e., players should be able to play the game via the console using the keyboard. There should be a command to print the state of each player (where they are on the board, how much money they have, which properties they own), a command to buy the property they landed on (if available), and a command to pass your turn to the next player. Events such as landing on a property owned by another player (and therefore paying the indicated rent), the bankruptcy of a player, etc. should be printed to the console when applicable.
Also required: the UML modeling of the problem domain (class diagrams with complete variable and method signatures, and sequence diagrams for important scenarios), detailed description of the choice of data structures and relevant operations: you are providing an initial design and implementation for the Model part of the MVC. Do not worry about any GUI yet.

- Deliverables: readme file (see explanation below) + code (source + executable in a jar file) + UML diagrams + documentation, all in one zip file.
- Deadline: Friday Oct 22nd. Weight: 15% of the overall project grade.

Milestone 2: GUI-based version (now you're adding the View and the Controller!) of the game. Display must be in a JFrame, and user input is via the mouse. You have freedom for other GUI decisions. Also required: Unit tests for the Model.

- Deliverables: readme file + design + corresponding tests + code + documentation, all in one zip file. In particular, document the changes you made to your UML and data structures from Milestone 1 and explain why.
- Deadline: Monday November 8th. Weight: 20% of the overall project grade.

Milestone 3: Additional features: houses, hotels, and special properties and squares such as: jail, "Go", railroad, utility. Also, the ability to use any number of "AI" players. The way the "AI" player plays is up to you, and it could simply involve hard-coded decisions and that's good enough for our purpose. But if you're really intrigued by investigating true AI, I'd recommend Reinforcement Learning as a technique: the AI player would keep playing against itself and update the relative weight of decisions over time based on whether it won or lost (or shorter term, whether its overall wealth increased or not).

- Deliverables: readme file + code + corresponding tests + refined design + documentation. The program must work robustly, and the code must be "smell-free" (we will be hunting for smells!). Make sure that you document the changes since the last iteration, and the reason for those changes.
- Deadline: Monday November 22nd. Weight: 30% of the overall project grade.

Milestone 4: Two more things: 1- Save/load features. You may use Java Serialization to achieve this. 2- International versions with custom street names, values and currencies. The customization may be defined in XML or JSON format.

- Deliverables: readme file + code + tests + documentation. Your project should be well packaged, and the program(s) should be easy to install and run.
- Deadline: Monday December 6th. Weight: 35% of the overall project grade.

Milestones must contain all necessary files and documentation, even those items that are unchanged from previous milestones. Missing files cannot be submitted after the deadline, no exceptions. Verify that your submission contains all necessary files (in particular, don't forget to include your source code!) before submitting on Brightspace. The "readme" file, listed as a deliverable for each iteration, is typically a short text file that lists and describes: the rest of the deliverables, the authors, the changes that were made since the previous deliverable, the known issues (known issues are graded less severely than undocumented ones!) and the roadmap ahead.
"Documentation" includes up-to-date UML diagrams, detailed descriptions of design decisions made, complete user manuals, and javadoc documentation.

Note that nobody is stopping you from working ahead of schedule! In fact, iteration i+1 will very often give you good insight into iteration i.

This is a **team** project. Each team should have 4 or exceptionally 3 members. A project's success obviously depends on contributions from each member! So divide the work and cooperate. **Each contribution (source code, documentation, etc.) must contain the name of its author**: we will use this to determine whether there is any significant difference in the quality and quantity of the contributions of the team members. If any such difference is detected, the individual grades will be adjusted accordingly.
You are expected to use Github to manage your project (version control, issue-tracking, etc...), but the deliverables for each milestone should also be submitted for marking on Brightspace. Please use a private Github team repository; Github provides this to student accounts; but we are also happy to create a private repo should you need one. A TA and/or the instructor will also be members of each of the Github teams, so that they can track your use of the tool, verify that all group members are contributing, and their feedback will be given by opening new Github Issues.

The marking scheme for each iteration will be comprised of: completeness of the deliverables, the quality of the deliverables, and, for iterations 2 to 4, we also look at whether you took into account the feedback you received from the TA in the previous iteration.

Copyright matters!

Note that Monopoly is a commercial product, and there’s copyright attached to it. It is one thing to use copyright material under “fair use” for educational purposes, it would be quite another to profit from someone else’s work by offering a copycat version. The law is complex on that front, so please exercise due diligence if you want to make your Github repo public once the course is over.