SYSC 3110 Project – RISK!

The goal of this team project is to reproduce a simplified version of the classic strategy game RISK. If you don't know the game, you can play "RISK: Global Domination" as a free download from Steam. The map and the rules can also be found on Wikipedia.

We will skip the initial phase where players normally take turns placing armies on countries, and instead we will start with a random allocation of armies. This corresponds to the "auto" setup in "RISK: Global Domination".

We will also skip the phase of drawing cards and using them to receive more armies.

But for the rest we will stick with the official rules, and the official map and countries. The number of players ranges from 2 to 6, and the corresponding initial number of armies is 50, 35, 30, 25, and 20 respectively, depending on the number of players.

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 the map (i.e., which player is in which country and with how many armies), a command to decide which country to attack from which country, and a command to pass your turn to the next player. Events such as the outcome of an attack, whose turn it is to play, the elimination 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 23rd. 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 9th. Weight: 20% of the overall project grade.

Milestone 3: Additional features: bonus army placement + troupe movement phase + "AI" player. As per the rules of Risk, at the beginning of a player's turn, the player receives reinforcement armies proportional to the number of countries held (total territories divided by 3, with a minimum of 3) and bonus armies for holding whole

continents (number depending on the size of the continent, see rules for exact numbers). The player must place these reinforcement armies before moving on to the attack phase. The troupe movement phase occurs after the attack phase; the player can decide to move any number of armies from one country to another "connected" country ("connected" meaning that there must be a path between the two countries that only goes through countries held by the player making the move). As for the "AI" player: any number of the players in the game can be assigned to be an "AI" player. The way the "AI" player plays is up to you. One possible way is to first come up with a function that assesses how "good" a given state of the board is (in terms of countries held, armies, etc.), that is called a *utility* function. In a game that involves some uncertainty due to dice rolls, one simple yet effective AI method is to make it choose the action that maximizes the *expected utility*.

- 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 23rd. 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- Custom maps. The custom map may be defined in XML or JSON format. Upon loading of a custom map, the program should be able to reject invalid maps, e.g. maps where certain countries or groups of countries are unreachable.

- 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 7th. 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 cuLearn.
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 cuLearn. 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 RISK 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.