IP2: Prototype *Automation: The Game*

In this task, you will build a Java program that implements the basics of a deck-building card game called *Automation: The Game* (ATG).

The goal of this assignment is to familiarize you with the mechanics of this game, and to give you more practice with writing complete Java programs from scratch (including unit testing).

Project logistics

Setup

You will work in the **same GitHub repository**, with the same development environment, from Individual Project 1.

Submission

1. Merge all changes into the `main` branch of your git repository, and push the changes to GitHub
2. Create a GitHub release of your repository
3. Submit the zip file from your release on Moodle.

Grading

Grading for this project will focus on the following areas and criteria:

- (40 pts) Whether the program functions as specified in the "Game Mechanics" section below
- (30 pts) Unit tests, including test quality (e.g. tests should make meaningful asserts and cover a meaningful range of cases)
- (20 pts) Code style: class/variable naming, good use of encapsulation, modularity in code (e.g. avoiding extremely long methods), effective commenting (e.g. meaningful comments explaining unclear or complicated sections of code; you are not required to exhaustively javadoc your code for this assignment)

Updated 1/16/25

- (10 pts) Whether the submissions adheres to the non-functional instructions (correct repository name, Java package/class names, properly formatted zip file submitted, etc)

Game Mechanics

The mechanics of this game are based on the popular card game Dominion (full rules, play online). We will start out by building a small subset of the mechanics, and add to them over the course of the semester.

Overview

Each player has their own deck of cards in the game. Over the course of the game, players can add cards to their deck by purchasing them from a shared supply. Some cards contribute Automation Points (APs) to the deck, and other cards give the player money to spend during their turns. On each turn, players draw a hand from their own deck, play cards to gain purchasing power, and purchase a card to add to their deck. The goal of the game is to construct a deck of cards that is worth the most APs.

Cards

For our prototype version of the game, there will be 2 kinds of cards: Automation cards and Cryptocurrency cards. In the full game, there are also "Action" cards, but they are not required for this prototype.

Automation cards have:

- A cost - the number of cryptocoins required to buy the card
- A value - the number of APs the card is worth at the end of the game

Automation cards in the game:

- Method x14 (cost: 2, value: 1)
- Module x8 (cost: 5, value: 3)
- Framework x8 (cost: 8, value: 6)

Cryptocurrency cards have:

- A cost - the number of cryptocoins required to buy the card
- A value - the number of cryptocoins the card is worth when played

Cryptocurrency cards in the game:

- Bitcoin x60 (cost: 0, value: 1)

- Ethereum x40 (cost: 3, value 2)
- Dogecoin x30 (cost: 6, value 3)

Note that after cards are played during a turn, they remain in the player's deck. This is why the cost of cryptocurrency cards is greater than their value - they will each be played many times over the course of the game.

Setup

Our prototype game will support 2 players, and the players should be automated (that is, once a game is started, the 2 computer players should play against each other with no human input).

Each player begins with a starter deck, consisting of 7 Bitcoins and 3 Methods. These cards are distributed to each player at the beginning of the game, from the supply.

Players shuffle their starter decks and deal 5 face down cards to make up their initial hand.

The starting player is chosen randomly.

Turns

Turns consist of 2 phases:

- Buy phase: player plays cryptocoins from their hand, and can buy up to 1 card using the value of the played cryptocoins.
    - Bought cards go directly into the player's discard pile.
- Cleanup phase: player discards their hand and all played cards, and deals a new hand from their deck
    - Before dealing begins, the previous hand should be added to the discard pile.
    - When dealing a new hand, cards are dealt from the player's draw pile until it is empty.
    - Once the draw pile is empty, the discard pile is shuffled, and becomes the draw pile.

Game end

The game ends when all Framework cards have been purchased. The winner is the player with the most Automation Points in their deck.

Other requirements

1. Your project must use Maven as the build tool (see Individual Project 1, part 5 for how to set up a Java project using Maven)

Project Guidance

It's highly recommended to play a game or 2 of Dominion online to get a feel for the overall game. This may help clarify any rules or specifications that aren't immediately clear. Post to the "Ask the Class" forum for any other necessary clarifications.

It might be tempting to create an in-depth, extensible design for this project that will easily support the full Dominion game rules in the future. However, you are encouraged to work iteratively and incrementally, and to keep your solution as simple as possible. Don't overdesign or overengineer your solution. Meet the requirements of this project as simply as you can. For example, if you find yourself trying to design a complicated, extensible class hierarchy for cards, or a complicated strategy for the automated players, you may want to simplify your approach.

Think of this project as a prototype of the game, to be used to decide if the gameplay is good enough to warrant further investment in development. You may not end up implementing this game at all, so overengineering at this point is wasted effort.

Resources

- You are encouraged to use code generation tools for this assignment. CoPilot with Sonnet 4.5 as the model is a good starting point (your GitHub Global Campus benefit gives you a fair amount of free usage).
    - Don't forget planning and review phases, as well as human review of the generated code.
    - Pay special attention to the test coverage - if the test coverage is good, you can have more confidence that the game itself is bug free.
- You can play the full Dominion game online here: https://dominion.games/ (free, but registration required)