Carbohydrates APCS 1 pd01

Jesse Sit, Xing Tao Shi, Joshel Xiedeng 2017-01-05

Final Project Proposal

**Monopoly Deal**

**General Rules:**

Monopoly Deal is a card game based off of the monopoly board game. The deck of 110 cards consists of 34 action cards, 28 property cards, 20 money cards, 13 rent cards, and 11 property wildcards. The goal of the game is to acquire three full sets of property cards in the property pile.

The game begins by giving each player five random cards from the deck. The remaining cards that weren't given out become the draw pile. Each turn, you draw two cards and then are allowed to play up to three cards. Money cards must be put in the bank. Property cards and property wildcards must be put in the property pile. Action cards can be put in the bank or played for their effects. Rent cards can be played to force other players to pay. You are only allowed to pay the opponents by cards in your bank or your property. If you have neither, you simply don't pay. Additionally, no change is given. For example if you only have a 4m property and they charge you 3m, you are forced to give up your property. However if you have neither a bank or property, you don't have to pay at all. At end of a turn, you can have at most seven cards in your hand.

**Basic Code:**

The layout of the game will be a **2d array** with alternating rows of banks and property piles of each player. The hand will be a single array. The deck will also be an array but it will be private so that the players are unable to view it. We will use a **random shuffling effect** similar to what we did in the slots assignment and then create a new variable to store the shuffled deck. When the game starts, it will use a user input to select between two to four players. Players will then take cards from the beginning of the deck array and the cards will be added to the end of the hand array.

The deck will be an array populated by instances of each subclass of the **superclass card.**

Each player's hand will be a separate array, sorted in order of property, wild property, rent, action, and money. All cards will have an instance variable that determines what the card ID is and this will be used to identify the card. This will also be used in order to **sort** the hand. There will be four subclasses of the card superclass which will be: property, wildcard, action, and rent. The only instance variable of the class card will be value (how much you can use the card to pay for) since it is the only variable that every card has in common. The class card will also have an accessor method for the variable value. Since money cards are so simple, they only have a value attribute so they will simply instantiate the class card. The class property will have the variable's name, rent value and set number and will have subclasses of colors. None of the color subclasses of property will have additional instance variables, they will have different values for the variables of class property. The wildcard class will have the variables color0, color1, and a boolean variable multi-color. Action cards will have instance variables name and description. Rent cards will have instance variables color0, color1, multi-color, and pay type. Each of these classes will also have **overloaded constructors** for when instantiating them.

The player order will be chosen **randomly** at the start of the game for the duration of that game. In order to cycle through the players, there will be in instance variable to count the total turns. The player who is the total turn module the number of players will be the current player and when they end their turn, the total turn will increment by one to move onto the next player. At the start of each turn, if the player has no cards, then they will draw five cards, else they will draw two. We will implement **try catch** in order to check if you have a card in the index chosen. An instance variable will be used to keep track of the number of moves made. When you use all three of your moves, you will be asked to end your turn. Type in "end" in order to end your turn. In order to use a card, you select the index of the card. If it is an action card, you will be given the option to use it or add to the bank. If you have more than seven cards at the end of the turn, you will be forced to select cards to discard until you reach seven cards. You will be prompted to pick the indexes of the cards that you want to discard.

At any point during your turn, if you think you've won, simply type "win" into the terminal and it'll check if you have three sets of property. This will be coded like this so that the game won't have to keep checking if the player has won so this will save execution time. Another approach that we could take

if this fails is creating an instance variable which counts up whenever you gain a full set of cards and when it reaches three you will be declared the winner.