Justin Phillips

cs362 w2020

Assignment 2

Refactor

The refactoring done in testUtility was a function called GetBoxes to create a set of boxes this function took nV as an argument and then returned box, also created a function supplyOrder which created a set of available cards and how much they cost. This function returned the supply_order. there was also a function which handled the supply of cards which was called randSupply this function took box as an argument then shuffled them to get us 10 random cards for the supply and then returned the supply. The last code that was refactored was the function that created the players that were in the game, it took the player_names s an argument then an array of players and then assigned them to the associated type of player. After this was complete, we returned the players.

Test Scenarios

For all test scenarios I switched the players to all to be controlled by the computer. I did this so I could see how the program would run in full with the scenarios I implemented. The test scenario for testDominion1 was that I changed the supply of copper, silver and gold to zero. It affected the game by eliminating the amount of currency that could be purchased. The game play went faster then I thought it would I ran the game many times and the least amount of turns in a game was 7 and the most was 18.

The test scenario for testDominion2 was that I tested what would happen if nV was set to 1 when there were more than 2 players. The bug that was created was that the first person that would buy a duchy or providence would end the game. I also tested what would happen if nV was zero the results were that the game would just end in a tie. This is because nV was used to set the supply of some of the victory cards. Since the supply was multiplied by zero it triggered the gameover code activate. This is significant since the game would end before it ever got started.

Part 3 Bugs

The bugs that I tried to create was a bug that would cause the game to never end because of eliminating the ability to purchase any currencies and the other bug I was trying to implement was a game that had little to no properties available The first bug much to my surprise the games went surprisingly fast. I guess the bug I was trying to implement wasn't successful since this was opposite of what I was trying to accomplish. The other bug was to set zero or 1 duchy and province that could be purchased. I did this by adjusting the value of nV to the corresponding values. My intention was to see what would happen if the game had these values. I was expecting the game to end early and that exactly what was happening. The situation where I had nV at 0 the game wouldn't even start it would just go directly to the end game results. The other situation where I set the nV value to 1 the game would start as usual but ended as soon as duchy or providence was purchased. This should have been expected since one of the ways the game will come to an end is when the last province is bought or when three piles are empty. (In a four-player game, it's over when four piles are empty.) I guess in the second bug it worked as intended.