April 8th, 2024

Worked on creating a function to check if a player has won the game or not.

| //Checks the scores of every player and prints if a player is about to win or has won **int** FarkleCore::CheckForWin() **const** {  **int** winnerPos = -1;  **int** winnerScore = 0;   **for** (**int** i = 0; i < numOfPlayers; i++) {  //Get current players score  **int** score = players[i].GetScore();    //If that score is a winning score and it is the highest score thus far  **if** (score >= 10000 && score >= winnerScore) {  //set the new winning score  winnerScore = score;  //set the winning player  winnerPos = i;  }  }  **if** (winnerPos != -1) {  //If the winner has been set and every other player has rolled again  **if** (winner.GetName() == players[currentTurn].GetName()) {  cout << players[winnerPos].GetName() << " wins the game with " << players[winnerPos].GetScore() << " points!" << endl;  winner = players[winnerPos];  **return** 1;  }  **else** {  cout << players[winnerPos].GetName() << " is about to win with " << players[winnerPos].GetScore() << " points!" << endl;  winner = players[winnerPos];  }  }  **return** 0; } |
| --- |

Moved global variables to a GameInfo header class as a means of organization. Will need to modify the current codebase to use GameInfo.h however.

Ended up splitting functions between the Game file and the Core file. While the game handles add and removing players, the current turn information, the scoreboard, and the dice; the core file handles “front end” operations.

April 10th, 2024

While reading over the rules again for the WriteToFile assignment I realized that the player needs to have the option to choose what dice to hold for points and what dice should be rerolled. I feel this will need a bit of trial and error to get right - but I am thinking of using one array to store the current dice rolls that player puts aside, then have another array for the current roll that is taking place. The current roll will be checked for point dice and if any are found a list of options will appear to allow the player to choose how many points they want to store in the event of another reroll.

Turns out there are more point variations for Farkle than what is listed on the given document. Might as well add them, but will need to change how the dice are calculated. Most likely will end up with a long if-else chain. Will give myself the challenge and see how it goes.

| Dice Faces | Points |
| --- | --- |
| Single 5 | 50 |
| Single 1 | 100 |
| Three 2’s | 200 |
| Three 1’s | 300 |
| Three 3’s | 300 |
| Three 4’s | 400 |
| Three 5’s | 500 |
| Three 6’s | 600 |
| Four of a kind | 1000 |
| Three pairs of Two | 1500 |
| 1 - 6 straight | 1500 |
| Four of a kind + a pair | 1500 |
| Five of a kind | 2000 |
| Two triplets | 2500 |
| Six of a kind | 3000 |

April 12th, 2024

Got something working with the new systems to allow the player to choose what set of dice to set aside and if they want to reroll the remaining dice will be used. In regards to the expanded point checking for the table above it appears to work after a while of tinkering with it, but of course it will most likely break when I don’t expect it. Point check consists of a switch statement checking how many dice have been rolled, and then from the greatest amount of dice rolled to the least check every possible combination of dice that can score points. Once a score combination is found it adds said combination to a dynamic multidimensional array - or an array that can be resized but the start of the new dimension is a displacement on the normal array. In this array a the key input to select the combination, code for what kind of combination, and the starting index in the dice array are stored. This is then read when the player selects the combination, where the selected dice are moved to the “side” array, points added, and current dice are set to 0. If multiple combinations are available the player can select from those, and if none are available then the program will back out to the main menu. In the event there are still combinations then a back out option is displayed to allow the player to not take score dice if they find it undesirable.

April 13th, 2024

Quite a few bugs with the logic checks when checking the dice faces for points, and some other crashes in regards to dynamic arrays - most likely because I do not fully understand C++ and the heap but after some research I was able to resolve that one. I am going to continue porting my previous functions to the new Game and Core setup I am currently working with.