*IN PROGESS, WILL FINISH TOMORROW MORNING (Game.cpp Line 295)*

*Exercise 1: Penalty Logic (Game.cpp)*

***Explained***

***This exercise …?***

[GameRunner.cpp]

[Game.cpp]

[Game.h]

1. [GameRunner.cpp] Code begins by creating a random generator and seeding it w/ a command line arg.
   * If there is only one cmd line arg (the default way of running the code), use the time as the seed, otherwise if a specific number is passed then use this as the seed.
2. [GameRunner.cpp] Next, a default “BasicBoard” object is created.
   * [Game.h] Default basic board object is a child of “Board” and passes up the number 12 to initializes the num\_places() array.
3. [GameRunner.cpp] Three “BotPlayer” objects are now instantiated passing both a str and a bool.
   * [Game.cpp] BotPlayer(str, bool) is derived from the “Player” object and is using a move constructor to steal the contents of “Player” and transfer them to “BotPlayer” in order to initialize the following values with the ones passed:
     + m\_name{str},
     + m\_purse{0},
     + m\_place\_idx{0},
     + m\_is\_in\_penalty\_box{bool}
4. [GameRunner.cpp] (vector<reference\_wrapper<Player>> players;) Next a vector called “players” is created to hold **REFERENCES** to “Player” objects as opposed to “Player” objects themselves.
   * All three bot players are placed into the “players” array using .emplace\_back() instead of push\_back()
     + Still not 100% clear on this. The only difference appears to be that emplace\_back() constructs the string literal directly inside the array as opposed to making a copy and then placing that at the end of the array.
5. [GameRunner.cpp] Now a “Game” object is created passing the previously created “board” object and our “players” vector that we just filled.
   * [Game.cpp] The game object instantiates:
     + m\_board(board)
     + m\_players(players)
     + m\_cur\_player\_idx(-1)
6. [GameRunner.cpp] Finally, we call the “.play()” function on our “game” object and this last line of code is apparently where all the magic happens.
   * [Game.cpp] The .play() function takes no parameters and starts by calling the \_add\_players() function.
     + This function runs a loop the length of m\_players.size(). Pausing here ,\_players.size() is a vector that is being passed an vector full of players. Technically, isn’t the size of ,\_plyers.size() only going to be 1???? Check this!!