Team HAWDOGS
Maxwell Vale, Jude Grodesky, Jackie Li

APCS1 pd02
1.4.18

Final Project Proposal
The HawDog Casino

Our project is an interactive casino, one where the user can choose from a wide variety of casino games. A person will be able to spend money to play various casino games, such as Blackjack, Poker, Roulette, Keno, Craps, and Baccarat. Many of the games will have a brief description of their rules and how to win the game. Depending on their skill or luck, they can lose or gain money, based on the game and how the player can win at the game they play. There are several game types included in the casino. Some are card based that require the player to deal with a deck of cards while they play. Other games will not and instead rely on other means to play the game, such as the slot machine and Roulette. The player can switch between the games after winning or losing them, and can choose to withdraw their earnings between each game to leave the casino with their current amount of money. Of course, the money isn't just going to go to waste once the player is done making big bank in ***The HawDog Casino….*** No of course not. Instead, once the player is satisfied with their winnings, they may visit ***The HawDog Shoppe*** in which they may cash in their chips and cash in exchange for exclusive digital trinkets and prizes.

Since several of the casino games included in ***The HawDog Casino*** include cards, we are going to create a class that to represent a real deck of cards that can be applied to every game. Since each game has different requirements for what the starting hands are, there will also be an abstract "hand" class that will be altered by each game so that the starting hand is tailored specifically to each game. Also what will be included is the usage of the Player's hand, and well as the presence of the Dealer's hand, if the game asks for it. Depending on the game, the hand will be altered for each to fit the criteria of the rules, and cards can be stored in both the dealer's hand and the player's hand. Both hands are classified under a greater class of Hand, which we can then modify based on the game, and both player and the dealer(if necessary) will inherit and use.

The HawDog Casino (revised)

Welcome to ***The HawDog Casino***, the casino destination for all AP Computer Science students and any other aspiring CSers! Luckily for you, the casino is always relatively empty because no other people will be there at any time besides you! This casino is chock-full of all of your favorite traditional casino games:

GAMES

CARD GAMES	DICE GAMES	RANDOM NUM GAMES	GAME MACHINE
➢ Blackjack ➢ Poker ➢ Casino War ➢ Baccarat	➢ Craps	➢ Roulette ➢ Keno ➢ Big Six Wheel	➢ Slot Machines

Game Process

1. Preconditions
    a. To make it a relatively fair game for all new CSers at ***The HawDog Casino***, the owners have restricted the starting money reserves to $100
2. SECURITY CHECK
    a. In order to verify that you are indeed the person on the provided ID Card, <Player> must provide name and DOB as indicated to be allowed to pass. Like most top casinos, ***The HawDog Casino*** is highly secure and age checks all that enter.
3. Playing
    a. Once verified that <Player> is of age to enter, <Player> may spend money as they like. <Player> may choose to visit Games or ***Ye Olde Shoppe.***
        i. <Player> can choose from a wide selection of games from four game types
            1. Card Games
            2. Dice Games
            3. Random Number Games
            4. Machine Games
        ii. Each game has a different Pay-per-Play cost (including various bet amounts depending on the game) and Average Probability of Winning, so <Player> is given the choice to decide where and how much to bet to play the game. As is normal for any casino, the higher the bet that the player bids, the greater the earnings will be, but the greater the loss as well.
        iii. Each game will come with a set of rules, describing how to play the game and the scenarios in which <Player> can win the game. Feel free to read these as much as you please for better understanding of each game.
    b. Player may remain in ***HawDog Casino*** until such Player has lost all of their money or decides that they have better things to do other than splurging (Of

course, depending on luck or skill, <Player> may make back money and last longer, or decide to keep winnings).

4. End of Game
    a. If (Player.exit() || Player.balance() <= $0 || Player.balance() < lowestGameCost()) { game ends and <Player> is escorted out by our humble, but extremely intimidating, security staff.}

Player Attributes

- Name:String
    - From user input
- Age:int
    - From user input
- Balance:int
    - Preset by default to $100 (may change)
    - Constantly checked as casino simulation is running:
        - Make sure that balance >= $0
        - Compare balance to the lowest playCost in ***The HawDog Casino*** and lowest itemCost in ***Ye Olde Shoppe*** to make sure that <Player> is still capable of spending money at the casino. There is a *strict NO LOITERING policy in place,* so <Player> should be knowledgeable enough to know when it is time to leave and avoid a hefty fine.
- getName():String
- getAge():int
- getBalance():int
- setName():String
- setAge():int
- luckBoost() [if possible] - for one run of a game, increase <Player>'s odds of winning; can be used a total of three times per visit to the Casino

Games

- playCost:int
    - Every time a particular game is run, the playCost amount is subtracted from <Player>'s balance. If <Player> does not have enough to play or place a certain bet, then a message will be provided to let <Player> know of their insufficient funds.
- about() - describe some basic game rules in case <Player> is a noob and doesn't know the ropes yet
    - <Player> will be prompted before playing to see if they would like to see game rules

- playGame() - runs the particular game chosen
- win() - specifies conditions required for winning

Card Games

- Each card game implements the Deck interface.
    - Specifies general methods that the card games should have.
        - deal(int) - passes specified number of cards to <Player>
        - shuffle()
        - burn(int) - throws away cards from use and not to <Player>
    - Since each game's dealing, use, and rules pertaining to the cards are different, the body for each method will be altered while still making sure that each game uses the method.
- win() depends on combinations of cards specific to each game

Dice Games

- roll() - naturally, the dice games require a roll method to roll the die

Random Number Games

- Unlike other games at ***The HawDogs Casino***, Random Number Games are completely governed by RANDOMNESS or , as gamers would say, <Player>'s RNG.

Machine Games

- spin() - activates the slot machine (currently the only game under Machine Games)
- In addition to a small win, jackpot() must also be included in these games.

What To-Do, What to do…

*YoRPG is a main model for the functionality of the Casino Simulation

- How to run the game
- Usage of class hierarchies

*Refer to use of Interfaces

*Slots class is a model for game processes and how games should function and appear within the overall simulation

Versions

1. Version 1
    a. Create first compilable version of the GameDriver class
        i. At first able to pick and play game of choosing
    b. Codify Deck and Card functionality. Test on a mock-up game for completeness/errors.
    c. Create all Game classes with appropriate class hierarchies, instance vars, and methods

    d. Winnings not affected by House Advantage (or House Edge, HE)
2. Version 2
    a. Create ***Ye Olde Shoppe*** and items in it
        i. Include ***Ye Olde Shoppe*** in places to go within Driver class
    b. Make the rules of each game more realistic and specific
    c. Some sort of visual for the game being played rather than all text
3. Version 3 (if time)
    a. All needed aspects of ***The HawDogs Casino*** are completely finished.
    b. Include other buyable items that may affect gameplay
        i. Drinks/Food
    c. Allow <Player> to use some sort of power-up
        i. Temporary luck boost
    d. Account for HE in the winnings for each game