Data Structures (2028C) -- Spring 2020 – Homework 2

Topics covered: Working with Classes

Homework due: ***Thursday, Feb 20 at 6:00PM***

Objective:

The objective of this homework is to create classes and implement those classes in a simple game.

Scenario:

You are creating a new electronic Roulette wheel type of game. In this game, the house and the player each get a ball to place in the wheel. The player will make a minimum bet then release their ball in the wheel to find where it stops. The player can then look at her results and can choose to double, halve, or keep the same her wager. If the wager is changed, the house now gets 2 chances to beat the player. The house will then release their ball(s).

The player wins only if her ball lands on a spot that has a larger value than the house (house wins all ties). If the wager is halved, the house keeps half of the bet but must win both of its chances. If the wager is doubled, the house only needs to win one of its two chances. The game is over when the player chooses to stop or runs out of money. We will simulate this game by creating an instance of the wheel for the player and for the house. This game will run from a command line/console window. This needs to be written using C++.

Example Scenarios:

Bet	Player	Bet Change	House	Result
100	12	None	10	Player total +100
100	12	None	13	Player total -100
100	12	Double	10, 10	Player total +200
100	12	Double	13, *	Player total -200
100	12	Halve	13, 10	Player total unchanged
100	12	Halve	13, 14	Player total -100

Requirements:

1. Create a wheel class. This should have the ability to change the range of values with a default of 1 to 10. It needs a spin method that will randomly assign a valid value for whatever ball is added to it.
2. Create a player class that includes an instance of a wheel class. The player should have a way to track their current amount of money.
3. Create a main method and any necessary support functions to enable a person to play the game until they lose or cash out. This should ask the player the number of values on the wheel at the start of the game (minimum 6, maximum 20).
4. Bonus: Create a hard mode. This is an option the player selects when starting the game. The hard mode will increase the range of values on the house's wheel every time the house loses and decreases the number of sides every time the house wins two turns in a row (minimum number of sides for house is the same number of sides as the player).

This must include a derived class of the wheel that includes an overload of the spin method that accepts a parameter with the player's result value.

Submission:
Submit all source code files and any required data files in a zip file. Include a write up as a PDF including:

- The name of all group members (minimum 2 members, maximum 4 members). Points will be deducted for groups of one or more than 4 unless prior approval was granted.
- Instructions for compiling and running the program including any files or folders that must exist.
- What each group member contributed. If the contributions are not equitable, what portion of the grade each group member should receive.

Submission should be submitted via BlackBoard.

Grading:

1. 20% - Wheel class is declared and defined appropriately. Declaration and definition are in separate files.
2. 20% - Player class is declared and defined appropriately. Declaration and definition are in separate files.
3. 50% - Program runs per scenario and requirements.
4. 10% - Bonus requirement works correctly.
5. 10% - Code is well formatted, well commented and follows a reasonable style.

If program fails to compile, the grade will be limited to a max grade of 50%. Note that with the bonus, the maximum grade possible is 110%.