Christopher Merrill
CS162
10/9/2016

Lab #2: Dice Game Design and Test Plan

INITIAL ANALYSIS (Brainstorm/Pseudocode)

MAIN:
Ask the user how many rounds they want to play. (Validate input)
Ask the user how many sides should be on player one's die. (Validate input)
Ask the user if player one's die is loaded. (Validate input)
Ask the user how many sides should be on player two's die. (Validate input)
Ask the user if player two's die is loaded. (Validate input)

Instantiate a Game class that has five parameters:
-player one:
- # of sides
- loaded or unloaded
-player two:
- # of sides
- loaded or unloaded
-total number of rounds

Run the play function within the game class.
end

GAME CLASS:
Function Required:

void play()
Do the following while the current round number is less than or equal to the number specified by the user:
Print the round number.
"Roll" each die.
Print the results of each player's roll.
Calculate the winner (the player with the higher roll).
Print the winner.
Add one to the winning player's score.
Add one to the number of rounds.

Print the winner (the player with the higher score)
End

Variables Required:
int roundNumber: holds the current round number
int totalRounds: holds the total number of rounds
int playerOneScore: keeps track of player one's score
int playerTwoScore: keeps track of player one's score

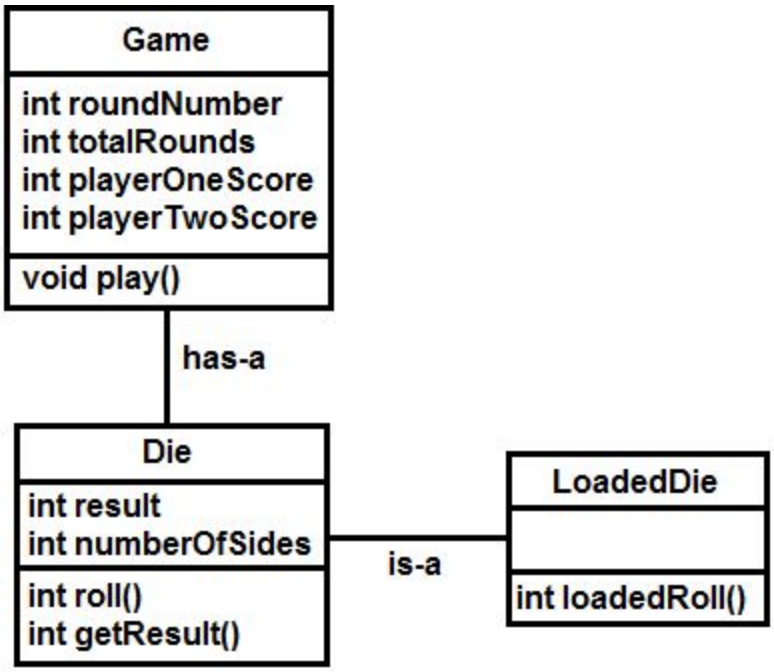
DIE CLASS
Function Required:
int roll()
Select a random number between 1 and the total number of sides on the die (using rand())

int getResult()
Gets the result of the last roll (used to determine the winner in the play() function above)

Variables Required:
int result: temporarily holds the result of the roll
int numberOfSides: holds the number of sides on the die

LOADED DIE CLASS
Functions Required:
int loadedRoll()
Select a random number between 1 and the total number of sides on the die (using rand()), but assure that there is a higher probability that the roll will result in the highest number.

CLASS HIERARCHY



TEST PLAN

Test	Expected Results	Results	Comments
Input too low (less than zero) on user entered integers (number of rounds and sides of die).	Asks the user to enter a value greater than or equal to 1 until a valid input is achieved.		
User specified round number is 1 (extreme low case).	Print the result of one round and immediately declare a winner.		
Die side number is 1 (extreme low case).	This particular die should roll a value of 1 every round.		
“Loaded” check: play through 100 rounds with two loaded die.	The results should favor the loaded value. (i.e. the highest side of the die should occur more often than not. Roughly 50%)		
Winner check for Player One: Give player 1 a one sided die, and give player 2 a ten sided loaded die. Play 10 rounds.	Player two should be deemed the winner (unless a statistical anomaly occurs, in which case try test again).		
Winner check for Player Two: Give player 2 a one sided die, and give player 1 a ten sided loaded die. Play 10 rounds.	Player two should be deemed the winner (unless a statistical anomaly occurs, in which case try test again).		