Christian Roccanova

CS 162

4/14/2017

The Dice War program will require 3 classes: Game, Die and LoadedDie. The program will simulate a game of War using dice. The number of sides of the dice and whether or not the dice are loaded will be set by the player. The program will need to roll the dice for the amount of rounds set by the player and will need to keep track of the victor of each round to determine a final victor.

The Die class will be the base class from which LoadedDie will inherit its behaviors and elements. The Game class will contain the menu, play and result functions. The game class will also contain a results array to store the rolls from each round. The menu function will first ask the user to input an integer for the number of rounds to play. It will then ask for the user to input the number of sides for each die to be used as an integer as well as whether or not the die is loaded, as a string. If the die is not loaded, it will then call the Dice function from the Die class to generate the die. If the die is loaded, then the loaded function of the LoadedDie class will be called to create a loaded die from the input integer.

The play function will loop for the number of rounds specified by the user from the menu function. It will contain playerScore variables for each player to keep track of their total scores. During each loop it will call the roll function from the Die class or the loadedRoll function from the LoadedDie class and will place the returns in the results array. It will then compare the rolls and increase the score of the winning player.

The result function of the Game class will print the player, number of die sides and whether or not the die was loaded for each player. It will then print the results array so that all rolls can be seen. Finally it will print the final scores of each player, compare them and then display who won.

The Dice function will take the given integer and use it to generate a die of N sides for that player. It will also contain a roll function which will take the number of sides as a parameter and will generate and return a roll using rand().

If the die is loaded, then the loadedDice function of the LoadedDie class will be called to create a loaded die from the integer given by the user. LoadedDie will also contain a loadedRoll function which will take the number of sides as a parameter and make a weighted roll by calling rand() three times and only returning the highest value rolled as its official roll for that round. It will do this by looping and comparing the current roll to the previous roll and updating the roll variable if it is higher.

The main function will merely serve to call the functions of the Game class as needed. First it will call the menu function to allow the user to select their options. It will then call the play function to run the game. Finally it will call the result function to display the results of the game. The majority of the function will be inside a do while loop to bring the menu back up at the end of a game to ask if the user wishes to play again.

Testing Plan

Input	Expected output
Any dice, 0 rounds	Error, asks user to try again
Any dice, negative rounds	Error, asks user to try again
0 sides, any rounds	Error, asks user to try again
Negative sides, any rounds	Error, asks user to try again
2 1-sided die, any rounds, any load	draw
2 6 sided die, any rounds, no load	50/50 distribution of wins
2 6 sided die with 1 loaded, any rounds	Distribution of wins should be skewed towards loaded die
1 4 sided die, 1 20 sided die, any rounds, no load	Distribution of wins should be skewed towards the 20 sided die
1 4 sided die, 1 20 sided die, any rounds, either die loaded	Distribution of wins should be skewed towards the 20 sided die
1 4 sided die, 1 5 sided die, any rounds, no load	Distribution of wins should be skewed towards the 5 sided die
1 loaded 4 sided die, 1 5 sided die, any rounds	Distribution of wins should be skewed towards the loaded die

Christian Roccanova
CS 162 Lab 2
4/14/17