Jackson Cummins

Professor Lusby

IT-312-H4706

21 April 2024

Final Project Documentation

**Problem Statement/Scenario:**

The LCR dice game consists of at least 3 players who each begin the game with 3 chips. The goal is to be the last player still in possession of chips. For each round of the game, one player will roll a number of dice corresponding to the number of chips in their possession. If the player has 3 or more chips, they will roll 3 dice. Otherwise, they will roll the same number of dice as chips in their hand. The dice has 6 sides each corresponding to a different action to be taken by the player. Rolling a 1 means the player must pass a chip to the player on their left while a 2 means the player must pass a chip to the player on their right. A 3 means the player must pass a chip to the center pot where the chip is no longer in play for the rest of the game. If the player rolls a 4, 5, or 6, nothing happens. In order to create a program to run this game, two classes are necessary. A player class will store information relating to each player in the game such as their name and the number of chips in their possession as well as functions that add or remove chips from their hand. A game display class will be responsible for executing the logic for the game such as organizing the players into a list, rolling the dice for each player, and figuring out if someone has won the game or not.

**Overall Process:**

Execution of the program begins in the LeftCenterRightDiceGame.cpp file where an instance of the GameDisplay class is created and then called the startGame function on the instance of that class. From here, the majority of the programs code takes place in the startGame() function of the GameDisplay.cpp file. From this function, the printRules() and addPlayers() functions are called which display the game rules to the console and then ask for the number of players participating and their names. After this, a while loop begins that calls the playRound() and checkForWinner() functions until the game is over. The playRound() function first determines whose turn it is and then determines which players are to the current player’s left and right. A message is then printed explaining how many chips the current player has and how many dice they will roll. The player’s dice are then rolled, and they lose chips if they roll either an L, C, or R. Finally, a scoreboard is printed at the end of each round displaying how many chips each player has remaining. The checkForWinner() function loops through the playerList vector and returns true if only one player is in possession of chips. When the checkForWinner() function returns true, a message is printed saying that the game is over and displaying the name of the winning player.

**Pseudocode:**

LeftCenterRightDiceGame.cpp:

CREATE instance of GameDisplay class myGame

INVOKE startGame() function on myGame

GameDisplay.cpp:

startGame() function:

CALL printRules() function

CALL addPlayers() function

WHILE game has no winner:

CALL playRound() function

CALL checkForWinner() function

END WHILE

PRINT game over message

END startGame() function

printRules() function:

CREATE fileReader object

READ game rules from a text file

PRINT each line of the rules to the console

END printRules() function

addPlayers() function:

READ number of players from user input

WHILE number of players is LESS THAN 3:

PRINT error message

READ number of players INT from user input

END WHILE

FOR number of players in game:

READ player name STRING from user input

PUSH player name to playerList vector

END FOR

END addPlayers() function

playRound() function:

CREATE currentPlayer INT variable equal to the index of the current player in the playerList vector

CREATE leftPlayer INT variable equal to the index of player before current player in playerList vector

CREATE rightPlayer INT variable equal to the index of player after current player in playerList vector

PRINT message stating the name of the current player and how many dice they will roll

WHILE currentPlayers number of dice is GREATER THAN 0:

CREATE INT variable diceRollResult equal to random number 1 – 6

IF diceRollResult EQUALS 1:

DECREMENT currentPlayer’s number of chips by 1

INCREMENT leftPlayer’s number of chips by 1

ELSE IF diceRollResult EQUALS 2:

DECREMENT currentPlayer’s number of chips by 1

INCREMENT rightPlayer’s number of chips by 1

ELSE IF diceRollResult EQUALS 3:

DECREMENT currentPlayer’s number of chips by 1

ELSE:

PRINT message saying nothing happened

DECREMENT currentPlayer’s number of dice by 1

END WHILE

END playRound() function

checkForWinner() function:

CREATE numPlayersWithChips INT variable

FOR each element in playerList vector:

IF player at current index has chips:

INCREMENT numPlayersWithChips by 1

IF numPlayersWithChips is GREATER THAN 1:

RETURN Boolean FALSE

END FOR

RETURN Boolean TRUE

END checkForWinner() function

**Methods and Classes:**

PLAYER CLASS METHODS:

Constructor Player(string) : creates instance of player class and assigns them a name

addChip() : increments the player’s numChips variable by 1

removeChip() : decrements the player's numChips variable by 1

getNumChips() : returns int variable numChips

getName() : returns string variable name

GAME DISPLAY CLASS METHODS:

startGame() : calls the other functions that execute game logic

printRules() : reads game rules from text file and prints them to console

addPlayers() : creates new players and adds them to playerList vector

playRound() : rolls the currenplayer’s dice to add and remove chips from the

appropriate players.

checkForWinner() : returns Boolean value false if there is not a winner and true if

the game is over.

getWinnerName() : NOT IN PSEUDOCODE. Returns string with the name of the

winner.

A computer screen with a black screen

Description automatically generated

A screenshot of a computer

Description automatically generatedScreenshot to verify that the program compiles and runs.

Screenshot showing output for printRules() and addPlayers() functions

A screenshot of a computer

Description automatically generated

Screenshot showing results for a few rounds of the game.

A screenshot of a computer

Description automatically generated

Screenshot showing results for the end of the game.