Project 1

Mastermind

Josh Drentlaw

2/1/2018

CSC-5-40652 OL

Introduction

The game I chose for my project is Mastermind. I remember playing it at a very young age with my dad, and having a lot of fun. I felt like it would provide a good challenge, while still being doable in the amount of time, and with the constraints we had. I have implemented a pretty complete version of the game with 2 human players, but have not begun to implement the game with a computer player yet. I hope to implement this portion of the game during the second part of the project.

Rules of the Game

The game is pretty simple: There are 2 players, a code maker, and a code breaker. The code maker uses colored pegs to create a sequence of 4 colors (different or the same) that is hidden from the code breaker. The code breaker has 12 attempts to guess the correct sequence, but after each guess, the code maker gives hints to the breaker about which colors they got right, and if they're in the correct position. The code maker has 6 colors to choose from: red, orange, yellow, green, blue, and purple. This allows for 1296 different patterns (6^4=1296). Points are awarded to the maker for each incorrect guess the breaker makes. If the breaker doesn't succeed after the 12th guess, the maker gets an extra point for stumping the breaker. The players chose an even number of matches, and switch off between maker and breaker roles. After the agreed number of matches is completed, the player with the most points is the winner.

Inputs and Outputs

The first thing that is output is a summary of the rules for the game. The first couple inputs ask for the number of human players, and for the number of matches. Both of these inputs are simple int's, and both are validated. Human players should be input as 1 or 2, and the number of matches should be an even, positive number. A match begins with the code maker opening the code-doc.txt file and entering a 4 character code like so: royg. They then enter 1 to move the program to the guessing phase, and behind the scenes, the code is read in from the text file. This was the best way I could currently think of to hide the code from the code breaker. Originally I had the console output several end lines, to shoot the code out of sight as quickly as possible, but it felt a little hack. The score is also displayed during the code entering phase, which will display 0-0 for the initial run. During the guessing, the output follows this pattern:

```
Guess i:   ← A for loop, loops through the 12 guesses, and i from that loop is used here
royb       ← This is the guess made by the code breaker
xxx-       ← Hints left by the code maker. x, o, or - are used.
```

When entering hints, the maker uses an to note the correct color and position, o to mark a correct color, but incorrect position, and a - to indicate that the color is not used in the pattern. I used a bit of logic I was planning to use for the computer to enter hints, to validate whether or not the code maker entered the correct hint or not, and will output a message if they did incorrectly input the hints.

Finally, the game outputs that the code has been guessed if the code breaker was successful, or if the number of attempts has been reached. In either case, it outputs the number of points the code maker earned during the current match. After all the matches have been played, it outputs the final score and the winner.

Pseudo Code

```
// HEADER
// INCLUDES

Int main
    // vars
    Cout << "RULES"
    Cin >> Enter # of human players
    If humans == 1
        Cout << "No computer"
    Else if humans == 2
        Cin >> even # of matches
    While matches > 0
        If maker == 1 // Same for 2
            Cout << score
                Cout << "enter code in file. 1 to continue
                Cin >> val
            Val = 0
            // open file and read code from it
            //validate code
        // Start guessing
        Cin >> guess
        Cout << hints
        Cout << points
        If i == guesses
            // Failure
        If chkHint == "xxxx"
            // Success
        Matches--
    Cout << game end
```