CIS 129 Final Project

This project was designed to give you practice building C# applications that use the various techniques and capabilities you learned about through the semester. The project is worth 200 points, and consists of a single C# application. Build the application using Visual Studio using a project. When you submit your work, compress the project into a single zip file within Blackboard. To decrease the size of the zip file, you can delete the `bin` and `obj` folders within the project folder if you wish.

Background

According to Wikipedia, "Old Maid is a Victorian card game for two or more players probably deriving from an ancient gambling game in which the loser pays for the drinks." Old Maid can be played with a standard card deck, but most people obtain a special card deck unique for the game. One of the more popular variants is shown in the figure below.

This deck consists of 21 cards made up of ten pairs of each card except the Old Maid card, of which there is only one (the image in the upper-left corner is not a card). The dealer deals all of the cards to the players, distributing them as equally as possible. Some players may have more cards than others; this is acceptable. Players look at their cards and discard any pairs they have. The game is then ready to begin.

Beginning with the dealer, each player takes turns offering his or her hand to the person on his or her left, making sure nobody can see them. That person selects a card without looking and adds it to his or her hand. This player then sees if the selected card makes a pair with any of their original cards. If so, the pair is discarded. The player who just took a card then offers his or her hand to the person on his or her left, and so on.

The objective of the game is to continue to take cards, discarding pairs, until no more pairs can be made. The player with the card that has no match is "stuck with the old maid" and loses. When playing with more than two players, the game is somewhat unusual in that it has one distinct loser rather than one distinct winner.

Functional Requirements

Your objective for this project is to build a simluation of the Old Maid game in C# with a variable number of players as determined by the user who runs the program. The program can be graphical or text-based. When the program starts, the user is asked for the name of each player. The user can determine how many players there are, but there must be at least two and no more than five.

While the game is played, the program indicates, for each player who has a turn, the card the player draws, whether the player keeps the card or discards a match, and how many cards the player has left. When one player is left with the Old Maid, the game is over and the program identifies the losing player.

Technical Requirements

You may build the program as a text-based or graphical-based application. You may use an object-oriented programming technique (which will earn you more points) or use sequential programming. Use any language constructs or techinques you like to implement your solution. You will be graded, in part, on how well you have designed your program. When finished, compress the project folder into a zip file and submit it within Blackboard.

Assessing Your Work

I will grade your project based on the following criteria:

- Does the program work? (40%)
- Does the program use the C# language capabilities in a way that maximizes the efficiency, length, complexity, modularity, and clarity of the code? (30%)
- Is the code commented and formatted, and documented to make it as readable as possible? (10%)
- Is the quality of the user interface intuitive and easy to comprehend by the user? (20%)

Suggestions for Getting Started

You are not required to implement a solution this way, but here are some suggestions if you're not sure how to begin: Create an object-oriented design with a `Card`, `CardDeck`, and `Player` classes. The Player should have a `List` of Card objects. The CardDeck should hold a List of Card objects and have the capability of randomly dealing the deck to a List of Player objects. The main program should create a List of

Player objects and a CardDeck object, deal the `Card`s to the `Player`s, and enter a loop that keeps running as long as more than one Card exists amongst all the `Player`s. For each iteration of the loop, the `Player`'s cards should be displayed, and a given Player should take a card from the next Player in the `List`. The program should tell the user if the `Card` is a match and discarded, or is kept. When one `Card` remains, the loop exits and the winner is announced.