Jeffrey Leonard

Chicago: The Card Game

CS 4900 DAY AA

May 7th, 2021

Introduction:

Chicago: The Card Game, is a variation of the popular card game, *Rummy 500*, that one of my family members had created. The goal of this project was to take that game and turn it into a game that can be played on a computer. There are two versions of Chicago currently playable; one is via a command line utilizing Rust, while the other choice is a GUI utilizing Python.

The game itself is very simple, as the hard part of the game is paying attention to what cards have been played, as well as what people are picking up from the discard pile. The goal of the game is to put down the cards that are required for that round, and then try to get rid of the remainder of your cards by either playing on yourself, eithers, or discarding your cards each turn.

Technologies Used:

The technologies that were used to develop *Chicago,* was the programming language Python, as well as the PyCharm IDE. My project heavily utilizes the *pygame* library, which is a set of Python modules that are designed for the specific use of video game development. The other libraries that I used include *pygame_menu, random,* and *operator*. I also used a resource called *pyinstaller* to create an executable of the game so that it can be played without needing to install Python or any of the libraries in an IDE. The system that I developed Chicago on includes an Intel Core i7-6700k CPU and a EVGA RTX 3080 GPU.

How To Play:

The rules of *Chicago* are straightforward. The overall objective of the game is to go out each round and maintain the lowest score possible. The game consists of two decks of cards, jokers included, for a total of 108 cards. You have runs, which are where any four or more cards in a row that share the same color and suit. For example, a 2, 3, 4, 5 of spades in red is considered a run. Next, there are sets, which are any three or more cards that are the same rank, but they can be any color. For example, a 3, 3, 3 of spades and hearts in red and black. There are 6 rounds of *Chicago*, where the first round is two sets, the second round is two runs, the third round is two sets and a run, the fourth round is two runs and a set, the fifth round is three sets, and the final and sixth round is three runs.

A round consists of three, four, or five players with eleven cards, the first person picks up a card, and if they meet the requirements for putting down for that specific round they can do so, otherwise you have to discard a card. During a round, everyone has the option to "May I". That means that if someone who is not up wants the card that was discarded, everyone has to agree to it and if so that person gets the card and a card from the shuffled deck, you can only have three "May I's" a round. During the last round, you have to "May I" at least once to meet the minimum requirement of cards needed to be put down, which is twelve cards.

The scoring system of *Chicago* is very simple. When someone goes out at the end of the round, everyone else has to count the number of cards in your hand. A 2 through 9 is five points, a 10, Jack, Queen, and King is ten points, an Ace is fifteen points, and a Joker is fifty points. A Joker is considered a blank card, where it can be anything the player decides it to be, but if put down with a set that Joker is considered dead and cannot be used anywhere else for the remainder of that round.

Fig. 1 – Example of a set

Fig. 2 – Example of a run

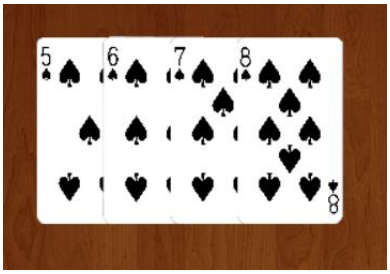


Fig. 3 – Example of a set with a joker

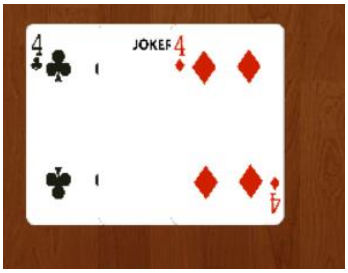


Fig. 4 – Example of a run with a joker

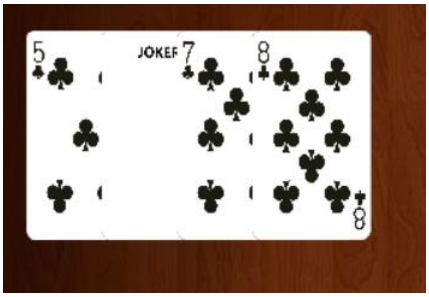


Fig. 5 – Main Menu

Fig. 6 – Rules Page

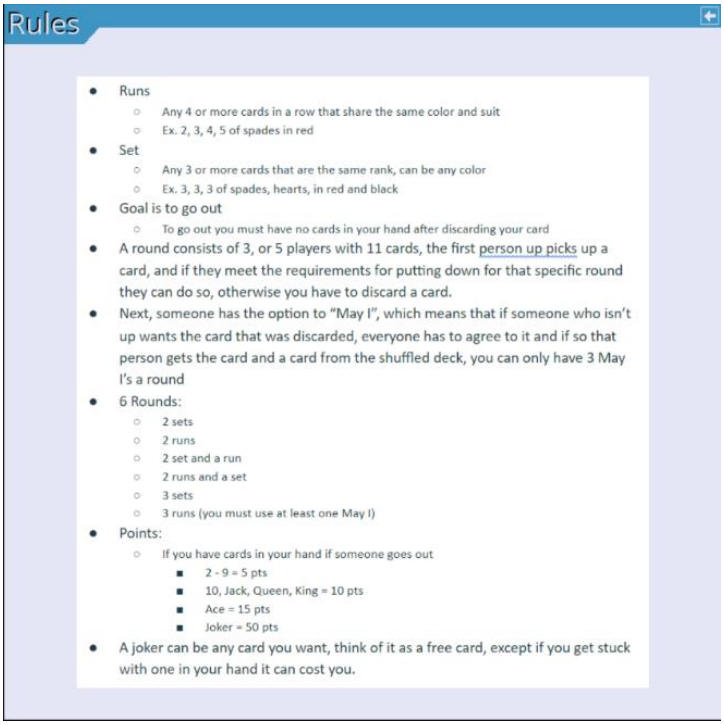
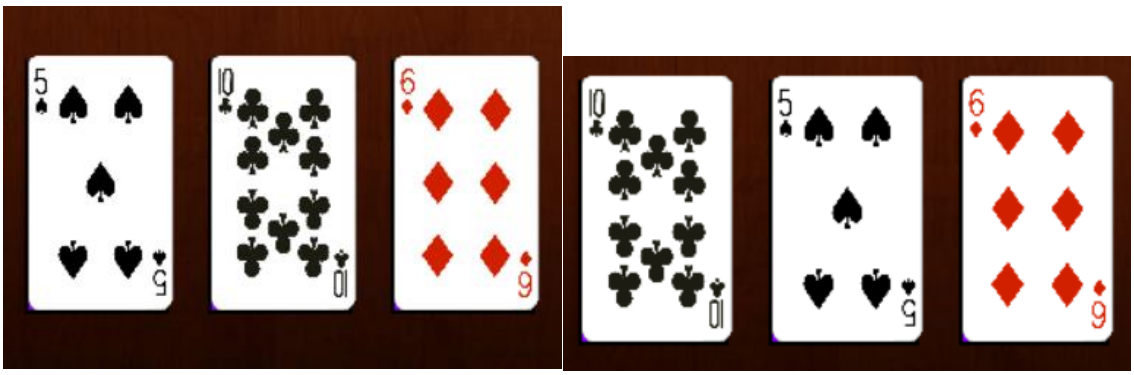


Fig 7 – Manual sorting of cards



- You click on a card, then on another card, and it will swap the positions of the card.

Fig 8 – Example of the Put Down action

Fig 9 – Example of the Append action

Fig 10 – Example of Scoreboard

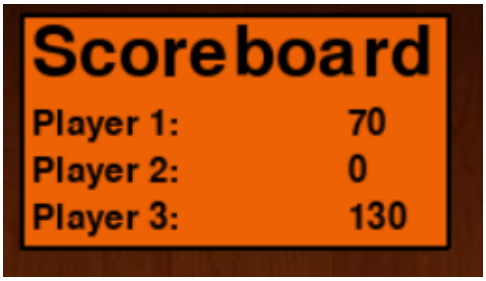


Fig 11 – Example of the game screen

Future Improvements:

There are some future improvements that I would like to implement into my game. The first, major improvement that I would like to implement would be multiplayer functionality. Currently, the game runs only locally, and it also does not help that you can see each other's cards that are in your hand. Once I implement a multiplayer aspect in the future, a major UI update would also be needed to incorporate the ability to add up to five possible players.

Another future improvement I had in mind was developing it for mobile devices. After spending time with Android Studio for another project, I realized that developing a mobile version of my game is an obtainable goal.

The next future improvement I had in mind would be auto scaling the resolution of the game based upon the monitor the player has. Currently, the game forces a 1920 x 1080 resolution, and in the future, I would like to add an option to either choose the desired resolution, or have it auto adjusted automatically without asking the user.

The last improvement I wish I could have implemented was a proper approval function of the cards that you are allowed to put down in each round. Currently, the user can put down any combination of cards down, regardless of what each round's rules are.

Problems Faced:

One major problem that I faced initially, was trying to get a GUI library working with the programming language Rust. Originally, the game was going to be developed in Rust, with the intent of converting a version of the game I had written that was playable via a command line, into a GUI. After trying multiple different libraries, I had come to the realization that I was not going to be able to develop the game in Rust. The language that I ended up using to solve that issue was Python. From previous experience with Python, I knew it had all the capabilities and libraries I needed to complete the project.

Another issue I faced was when I was creating the executable for my card game. Initially, I had tried to keep it to one file, but as of now you have to download a zipped file, with an images folder, a font file, and the executable itself in order to play it. I had to remove the main menu also due to the software I used to convert it to an executable kept giving errors related to the main menu.