**Hanabi Preliminary Specification**

**Project Description**

Hanabi is a cooperative card game for 2 to 5 players, designed by Antoine Bauza.

There are 60 cards in the game. There are five colors of cards and one set of rainbow cards. In each color, there are ten cards, numbered 1, 1, 1, 2, 2, 3, 3, 4, 4, and 5. Also in the game are 8 clue tokens and 3 lightning tokens. Each clue token has a light side and a dark side, and each lightning token has a blank side and a lightning side.

When setting up the game, all clue tokens are turned light side up and all lightning tokens are turned blank side up. The cards are shuffled. The rainbow cards are for the advanced game and may be omitted. Then, each player is given a hand of cards. For 2 or 3 players, there are 5 cards in a hand, and for 4 or 5 players, there are 4 cards in a hand. The rest of the cards are placed face down in a draw deck. There is one "pile" made for each color in the game (5 normally or 6 with rainbow cards), although these piles are all empty at the start of the game. One player is selected to go first.

Each player holds their hands backwards, so that each may see the others' cards, but not his or her own.

On one's turn, one has three options: (a) give a clue, (b) discard a card, or (c) try to play a card. Play then passes to the player to one's left.

To give a clue, a clue token is turned from its light side to its dark side. If there are no clue tokens with their light side up, no clue may be given. A clue is then given to a single other player. A clue is given by pointing out all cards that match a given number or color, of the clue giver's choice. If the clue giver gives a clue to another player about 4's, for example, the clue giver points out all 4's that the other player has in their hand. Rainbow cards match every color for the purposes of clue giving. The clue giver cannot lie, and must point out all cards that match the clue.

To discard a card, one simply puts a card from their hand into the discard. The discard pile is open, and players may see what has been discarded. When a card is discarded, flip one dark clue token to its light side, if possible. The player who discarded then draws a new card from the draw pile into their hand.

To play a card, one chooses a card from their hand that they will try to play. If the card is the next number is sequence for the pile of its color, then it is successfully played, and goes onto that pile. Otherwise, it is discarded, to clue token is flipped, and a lightning token turns to its lightning side. Each pile must begin with a 1 of its color, then a 2, then 3, and so on up to 5. If a 5 is successfully played, then a clue token is flipped from its dark side to its light side as a reward. Rainbow cards form their own separate pile, even though they match all colors for clues. After any attempt at playing a card, the player who played it draws a replacement card from the draw pile.

If all three lightning tokens are flipped to their lightning sides, then the players collectively lose.

When the last card is drawn from the draw pile, every player, including the one who drew the last card, gets one last turn. Then the game ends. Every player should have one less card in hand than when the game began.

The group's score is the number of cards that the group managed to play successfully. If 18 cards, for example, are on the table, then the score is 18. The number of clue tokens or lightning tokens flipped is immaterial. 25 is a perfect score, or 30 if playing with rainbow cards. The goal is to achieve the highest score possible.

**Project Purpose**

To provide users interested in the card game Hanabi a way to play the game with other players through the internet with a software that is user friendly and easy to install and operate.

**Project Details**

­I. Structural Design

Our structural design decisions are summarized in Table 1.

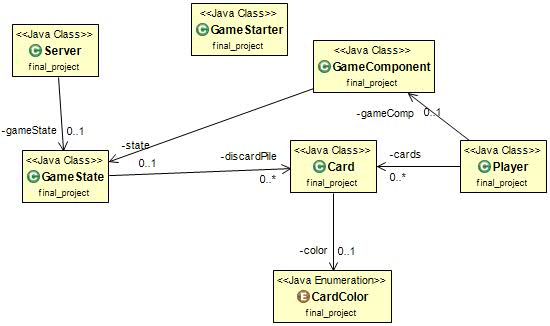
|  |  |
| --- | --- |
| **Data** | **Data structure** |
| Hands | Arraylist of lists of Card |
| Clues | List of boolean arrays |
| Deck and discard pile | Arraylist of Card |
| Color of card | Enumeration of colors |
| Players | Arraylist of Sockets |

**Table 1. Structural design decisions for Hanabi**

The hands for each player are stored in an ArrayList containing a List of Card for each individual player in the game. The clues are stored in a List containinga boolean array for each Card. Each boolean array contains values for various possible colors or numbers the Card can take on. Separate ArrayLists are used to keep track of the deck and the discard piles and the players on the server side. Finally, an enum is used for colors of cards.

II. Object-Oriented Design

Figure 1 shows the class diagram for Hanabi in Unified Modeling Language.



**Figure 1. UML diagram for Hanabi**

The Server class implements the server part of the network. It manages data transfers from the different clients. The GameState class holds data for the current state of the game such as the players’ hands, the state of the cards in the hands, and any knowledge about the cards. GameComponent draws the state of the game from a certain player’s point of view. Each player has a GameComponent. GameStarter contains the main for starting the game.

III. Testing

Proper testing will be achieved by utilizing a JUnit 4 test suite environment. The test suite shall test all unique possible combinations of input and output of each class and method.

**Project Objectives**

To design an intuitive graphic user interface that presents information about the state of the game, and to efficiently make use of system and network resources.

**Project Constraints**

The project will be subject to time constraints of a deadline at June 3rd. The team may not be able to implement more complicated features that may require more time.

**Project Milestones**

Table two shows goals and corresponding dates that the team wishes to achieve the major goals by.

|  |  |
| --- | --- |
| Date | Goal |
| Week of May 6 ~ 10 | Conclude the planning phase. Finish planning the GUI, how classes will interact with each other, and how data would be efficiently transferred through the network. |
| Week of May 13 ~ 17 | Start and complete the Graphic User Interface and all client side functions. Start testing and debugging the client side code. |
| Week of May 20 ~ 24 | Finish all client side details. Start coding server side data transfer algorithm and testing the code. |
| Week of May 27 ~ 31 | Double Check functionality of program, and add or remove any feature or functions as the team sees fit. |

**Table Two -** project milestones