***Section 1 – State the purpose of your project/sub-system***:

This project is a one round 4-player Blackjack Game. Blackjack is a popular card game, this game is design for people who like to play card game, or wanted to practice how to play BlackJack. Playing this game can kill time, It' a entertainment that can improve your playing skills and computational ability.

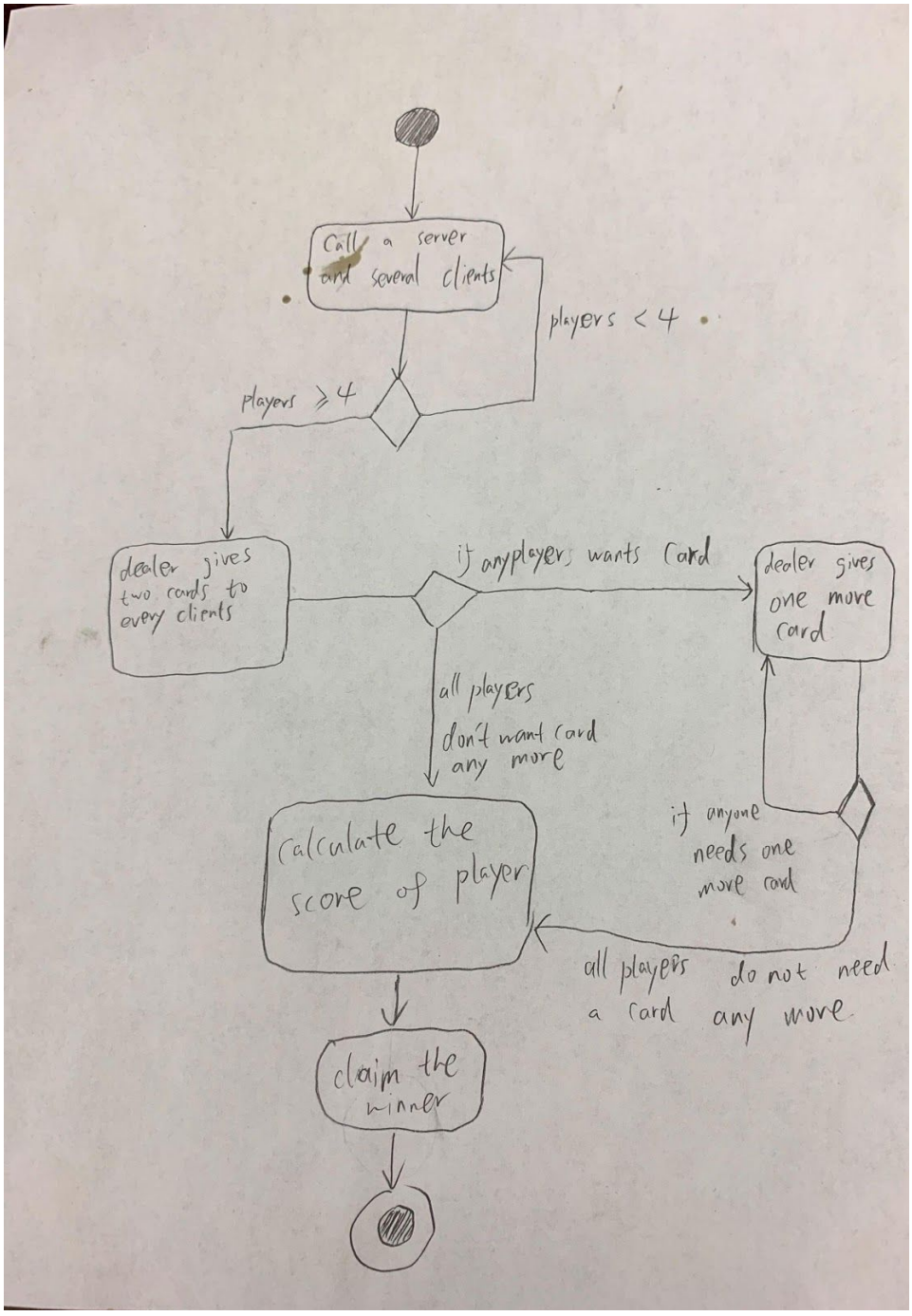
Each player will play against dealer. The code will be written in JAVA. The elements of the game, such as the cards in hand, the cards in deck, the client/server communication and the enable instruction, etc. will be assigned to different group members. The communication will base on the socket package from JAVA. When a player connected to server, the window will display the port number and IP address. Once total of four players connected to the server, the game will start.

In each round every player will get 2 cards, as well as dealer. The second card that every player and dealer has will be displayed in text area. Then player can choose to get another card, or pass.The additional card that player gets will also be displayed in text area. Then there is a AI algorithm for dealer.

***Section 2 – Define the high level entities in your design***:

The three most important objects in this project are BJServer, BJClient and NetworkConnection, because these three objects are doing the most reaction with each others. Other objects are more likely associate with server or client, such as Card, Deck and Player etc. BJServer and BJClient are doing most reaction with user, NetworkConnection is a multiple threads link with server and clients. Overall the connection between server and clients are based on passing string.

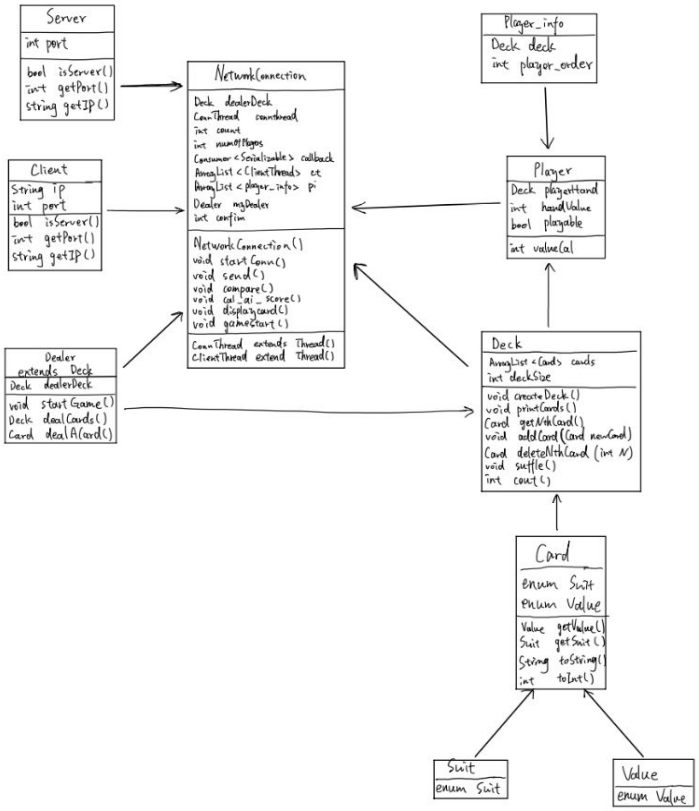
At the beginning, once when the server is turned on, server is able to accept clients. When there is total 4 clients are accepted into server, the game will start. Then dealer will give each client two cards, and one card will be hidden. If any player needs a card, dealer will deal one card to that client, and inform everyone what was that card. Every player can keep getting cards until they have total value above 21 points. If any player does not want any more cards, they can click submit, then he is no longer getting new card, and his total value will be recorded. Once when every player have submit or explore by over 21 points. Then result of score will be calculated and displayed.




***Section 3 – For each entity, define the low level design***:

There are some classes such as Suit, Value are enum class, these classes contain the card suit and value, these build the basic Card class data member, and there are methods that associated with card class to get access to the card. Then Cards build the Deck class, also

methods are associated with deck to get access to the data, such as getCards(), getDeckSize(), getNthCard(), and deleteNthCard(). After that, there are three different classes are sharing same deck, which are Client, Server , NetworkConnection and Player. Player class contains a deck of cards on hand, total cards value, and their availability. There is also a helper class keep track of players information called Player_info. Player_info contain their hand cards and the order of each player. In this project, dealer is act like enemy and game judge, which means Dealer class is almost identity with Player class, and some extra function such as startGame(), and dealCard(). Server and Client classes are more like container that hold the basic information for server client, such as IP number and port number. Now, the biggest class that does the most of work is NetworkConnection, this us the class create a deck of cards, then reference to other classes, also a ArrayList of ClientThread and Player-info, and the data between client with server. Once when there are total 4 clients connect to the server, the NetworkConnection will call gamestart(), then each player will have two cards, and on the text area, each player also knows one of two cards from other player and dealer. Then player can select either draw(), which dealer will give you a new card, then compare() will be called, if the total value of cards is over 21, then this player is out of our table. Otherwise, player can keep getting cards until his total value of cards gets close to 21.



***Section 4 – Benefits, assumptions, risks/issues***:

1. Base code for Blackjack
2. Playable for a single player
3. Communication by using thread class
4. Logic flow of how does each player play
5. Bookkeeping that keep track of each player

First of all, for the first time that working on thread, there are so many challenging we have overcome it. For the project, we are so glad, we did not have so much issus, and we have overcome a lot of problems. The logic flow and communicate between server and client was a big challenge for us. During the building this project, we have faced many issue, especially using thread send data back and force, we are not sure if thread can control the GUI, so we made a text area, and only two button while playing, which I think is a good way to handle all the different situation by just looking at the text area.

***Conclusion***

The hardest part of writing a design document has nothing to do with the writing. The difficult part is working through a logical design before you get to coding. Once you have a vision of how the objects and entities are arranged, writing the details is easy. In addition, it should not require anything more than a word processor and a simple shape painting program.