Group 3

Preface  
 This document specifies an electronic version of Chinese Checkers. Anyone with a basic knowledge of computers and board games is able to read this document.  
  
Introduction  
 This system is a game of Chinese Checkers implemented as a computer program, using the Java programming language. Chinese Checkers is a strategy board game which can be played by two, three, four, or six people, and can be both played individually or with partners. The object of the game is to race pieces across a hexagram shaped board into the triangular shaped base across from your starting position. There are few computer implementations of Chinese Checkers. We believe choosing a slightly less common board game coupled with the fact that we will implement network support will allow us to stand out from other games that are easily found online. The implementation of network support will likely be the most difficult aspect of our game to create, but it is also the most important aspect. One of the key differences between games like chess and Chinese Checkers is the number of players. Allowing players to play against other players across the Internet with up to five other players is a unique aspect of this particular electronic board game.  
  
Glossary  
 AI – Artificial Intelligence – Computer program that appears able to think for itself  
 UI – User Interface – What is seen on screen, the board, pieces, and back drop  
   
User requirements definition  
 Chinese Checkers is played using the board in the image below. It can be played with two, three, four, or six players, in teams of one, two, or three. For the six player game all triangles are used as starting areas. For the four player game, only two pairs of opposing triangles are used, for the two player game, only one. For the three player game three equidistant triangles are used. To start the game, each player is assigned a color from among the starting areas and the 10 pieces of that color are placed in the corresponding triangle.



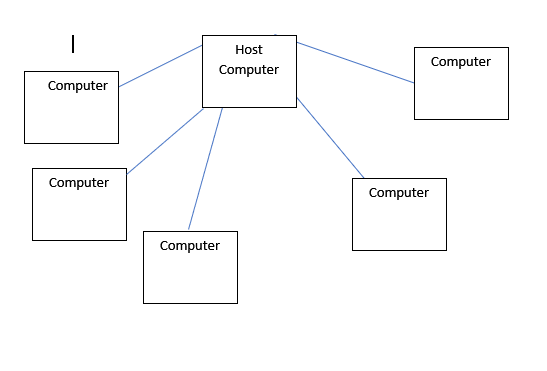
Credit: Denise Krebs

Link: https://bit.ly/2MVICOr

A team wins once all of the spaces in the triangle(s) opposite to their starting position(s) are occupied and at least one of the spaces in each opposing triangle is occupied by a piece of a different color than the triangle. A game ends immediately when a team wins.

Starting with a randomly determined player and proceeding clockwise, each player takes a turn, during which they must either: 1) Move a piece to an adjacent space or 2) Jump over a piece adjacent to it into the space directly behind it and then jump again in the same manner as many times as they wish. In addition to being able to play with users on other computers, players will also be able to play with up to six AIs. They will also be able to select the composition of each team; for example, they could play in three teams of two human players, or two teams of one AI player each and one human player, or even watch a game of six teams of one AI player each. In our electronic version of Chinese checkers, the non-functional requirements include a fast response time to player input, solid connectivity, and quick yet smart AI.  
  
  
System Architecture

The program will be written mostly in Java. We will use Swing to make the graphical model and UI. There will likely be three main screens which contain all other graphical model and UI elements: one for the new game menu, one for the lobby, and one for playing the game. In general the code concerning the game state will be kept separate from the code concerning the graphical model which will be kept separate from the code concerning user input events. The networking will use a client-server architecture over TCP/IP.  
  
System Requirements specification  
 In our computerized version of Chinese Checkers, the non-functional requirements include a fast response time to player input, a fast, user-friendly, and stable Internet connection method (given that the user’s Internet connection is stable), and an AI which chooses its moves within 5 seconds and makes reasonable and legal decisions. Responsiveness is a crucial part of the electronic board game experience. Whenever a player issues an input, the game must recognize it almost immediately, assuming the players’ Internet connections are not problematic. When a player is disconnected, the other players should be notified, and the program should shut down shortly thereafter. The functional requirements in our game include its UI, AI, network features, piece movements, rule enforcement, board layout, backdrop, board design, turn indicator, mouse controls, piece color, and creating a lobby to manage players going into the game. On a player’s turn they are prompted to click a piece and then prompted to click where they wish to move that piece. If the second click is an invalid move or is not on a space of the board, the piece is deselected and they are prompted to try to move again. To start a game the host must click the create lobby button, while other players must type in the address of the host and click the join lobby button. After a player clicks one of those buttons they enter the lobby screen. At the lobby screen the host can assign players to teams using combo boxes, choose the game mode, and click the start game button to begin, which will do nothing if the setup is invalid.  
  
System Models  
 Dataflow Model



System Evolution  
 This game will be designed with the intention the user is going to play it on desktop or laptop computer. It will not be designed to be played on a smaller screen such as a phone. Only being able to play this game upon a laptop or desktop computer will significantly limit the audience so porting our game to mobile may be something to consider in the future. Our core goal in this project is to create a hexagram shaped board which will allow the user to move pieces in accordance with the movement rules of Chinese Checkers. The game we create will allow the user to play against an AI, against others on the same computer, or against others online. Stretch goals we have for this project include creating different difficulty settings for the AI, implementing optional variant rules, making the game look exceptionally attractive and realistic, and  adding sound effects to fit the gameplay.

Preface: 5/5

Your preface is very good.

Introduction: 18/20

Your introduction explains the project well; however, you have some inaccurate discussion of how few Chinese Checkers games are available on the Internet. There are many, but few allow for network play.

Glossary: 5/5

Your glossary was short, but there were not many terms you needed to define.

User requirements definition 14/15

Your user requirements definition explained the game of Chinese Checkers very clearly. Some discussion of the GUI (buttons or menus to quit the game, perhaps a chat function?) would have been useful.

Of course, your group has a tremendous advantage in that you were trying to describe something that already exists.

System architecture: 9/10

Your system architecture was solid. However, you use statements like “game state” without being clear what that state includes.

System requirements specification: 13/15

Your system requirements specification covered most of the details. A UI mockup might have been useful. Discussion of UI and whether the game supports windowed, full-screen, or both modes should have been covered.

System models: 4/5

Your client server model is reasonable, but in actual fact, computers may communicate directly with each other. The host computer will probably share all IP addresses with all the other computers. Text explaining the model would have been helpful.

System evolution: 15/15

You discussed most of the important issues here.

Spelling, grammar, and style: 9/10

Your spelling was correct except for an instance of British spelling. You had a comma splice. Please be consistent with your spelling of “Chinese Checkers.”

Total: 92