

Project Name: Mancala

Team members:

Anjan Kumar Patel (140100089) (Team Leader)

Pulkit Ghoderao (140260009)

Harshank Shrotriya (140260014)

CLTA:

Arushi Jaiswal

Course Instructors:

Prof. Deepak B Phatak

Prof. Supratik Chakraborty

1.Introduction :

1.1 About Mancala

Mancala is a family of board games played around the world, sometimes called 'sowing' games, or 'count-and-capture' games, which describes the gameplay. Some popular mancala games are: Bao la Kiswahili ,Chisolo, Congkak, Kalah, Oware, Toguz korgool.

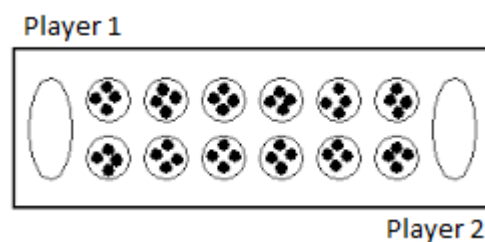
1.2 Kalah

Objective: The object of the game is to capture more seeds than one's opponent.

Rules:

1. At the beginning of the game, three or four seeds are placed in each house.
2. Each player controls the six houses and their seeds on the player's side of the board. The player's score is the number of seeds in the store to their right.
3. Players take turns *sowing* their seeds. On a turn, the player removes all seeds from one of the houses under their control. Moving counter-clockwise, the player drops one seed in each house in turn, including the player's own store but not their opponent's.
4. If the last sown seed lands in the player's store, the player gets an additional move. There is no limit on the number of moves a player can make in their turn.
5. If the last sown seed lands in an empty house owned by the player, and the opposite house contains seeds, both the last seed and the opposite seeds are captured and placed into the player's store. Thus ending the players move.
6. When one player no longer has any seeds in any of their houses, the game ends. The other player moves all remaining seeds to their store, and the player with the most seeds in their store wins.

It is possible for the game to end in a draw, with equal seeds captured by both players.



2. General

2.1 Selection of Topic

After considering the possible choices the team decided to build upon basic ideas taught in the introductory course on computer programming and extrude them to a higher level. This could successfully be achieved by a project on the game of Mancala, which would be unique in itself and which would incorporate most of the ideas discussed in the classroom.

Also, this would mean learning graphics in C++, which would be a good learning experience.

2.2 Purpose of the Project

This project is to be submitted in partial fulfilment of the course requirements of CS 101: Computer Programming.

2.3 Project Objectives

The team aims to transform the ancient board game of Mancala into a computer game.

The game must remain in its pure form as far as possible.

To include graphics in order to give a gaming experience which is close to physically playing Mancala on a board.

To provide maximum Entertainment!

3. Distribution Of Works.

3.1 Gathering and Outsourcing Information.

To collect information about game strategies, useful code samples and other resource material over the web and enlightening other team members about the same.

Liability of: All members.

3.2 Arranging Team meets and recording the minutes therein, Serving as a bridge between TA and the Team.

Liability of: Anjan Kumar Patel.

3.3 Incorporating Graphics

To study the graphics related issues of the project and generating corresponding codes.

Liability of: Harshank Shrotriya

3.4 Designing of specific functions.

With initial discussion, use of the following functions was identified.

3.4.1 InitializeBoard()

3.4.2 SelectStartingPlayer()

3.4.3 SyncBoard()

3.4.4 GetPlayerMove()

3.4.5 ExecuteMove()

Many more functions shall be identified subsequently.

3.5 Preparing final Draft of SRS document, User manual, Project Report

Liability of: Pulkit Ghoderao

3.6 Sample Coding

Liability of: Distributed among all members.

4. The Road Ahead:

As good programming practice requires; the problem statement, basic functionality and general algorithm have all been identified (refer: SRS document attached).

The job left is to complete final coding, include the best possible graphic options and provide a relatively good gaming experience.

In order to simplify the algorithm the team plans to modify a few rules subsequently.

For example: It seems an option, to change rule 5 above as

If the last sown seed lands in an empty house owned by the player, and the opposite house **may or may not contain** seeds, both the last seed and the opposite seeds are captured and placed into the player's store. Thus ending the players move.

Still efforts will be made to maintain the game in its pure form as far as possible.

5. The Road Traversed :

In view of the previous assertions, the team decided to finally exclude rule 5 above. It was generally agreed that the game would be far more simpler and code far more elegant.

Also, it was decided to stick to material covered in the course itself, the only exception being the use of simplecpp graphics.

Thought was given to include a circular queue but it would make the program clumsier. Hence, a simple one dimensional array was used.

Subsequent modifications to the two player version were made as and when needed and are stated in the SRS document.

The inclusion of single player version by the use of the `getbestmove()` function proved to be an important breakthrough.

6. Individual Contributions.

6.1 Harshank Shrotriya

6.1.1 Static graphic initiation and 'Playing' with simplecpp graphics.

6.1.2 Creating valid hole click function.

6.1.3 Creating complete code to simulate value change in the array.

6.1.4 Merging game over and free turn function.

6.1.5 Extrapolating two player to single player version with the help of `getbestmove()` function.

6.2 Anjan Kumar Patel

6.2.1 Coordinating between members and member functions(functions made by each member)!

6.2.2 Code for checking free turn.

6.2.3 Code for checking capture.

6.2.4 Initiating dynamic graphics(animation).

6.2.5 Game over function.

6.3 Pulkit Ghoderao

6.3.1 Code for displaying elements of the array on canvas.

6.3.2 Dynamic memory allocation, de-allocation and book keeping for the same.

6.3.3 Dabbling with files to include high score function.

6.3.4 Creating the `getbestmove()` function.

7. A Hurdle.

The `getbestmove()` function can be also called recursively playing the best move of the player and the computer for the successive turns to come. But this can cost a lot of time and processing memory. Hence, it was noted that the 'deepness' of the function should be limited according to the processing speed of the machine in use.

8. Optimisation

Optimisation and extension of the project can be easily done given sufficient amount of time and effort.

Below mentioned are some of the topics that this project could cover :

a. Save Game option to save the current status of the game and resume play from that point.

b. Including better graphic packages to make the gaming experience more interesting.

c. Needless to mention, the `getbestmove()` function needs to be updated for better and better performance.

9.Acknowledgements(for the entire project.)

a. Course Instructors

Prof.Deepak Phatak and Prof.Supratik Chakraborty.

For their attention and early replies to doubts on Piazza forum.

b.CLTA

Arushi Jaiswal.

For giving suggestions as and when needed.

c. Prof.Abhiram Ranade

The book An Introduction to Programming through C++, McGraw Hill Education, 2014, by Prof. Abhiram Ranade was referred. Also Simplecpp Graphics developed by him were extensively used.

d.Piazza (Q & A platform) www.piazza.com

e.Wikipedia- The free encyclopedia

For game rules and miscellaneous information.

