# Let's Play Mancala


**Amal Tidjani, Meckila Britt, Lily Westort**

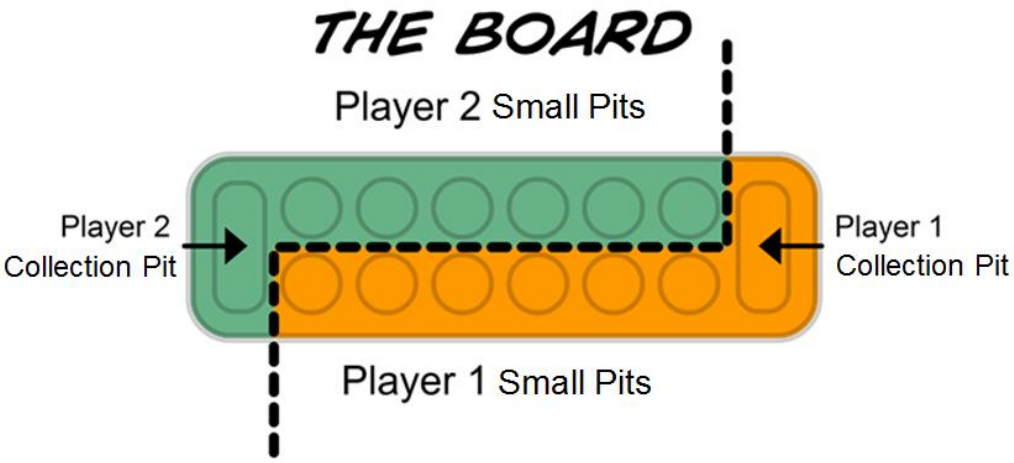
**CS230 Final Project**

**5/18/15**

# User Manual

***Mancala Instructions:***

This program executes a virtual version of the popular board game, Mancala. Before delving into the features of the GUI/Game, it is important to first establish the rules by which Mancala is played. Played between 2 players, the objective of the game is to collect the most pebbles in one's designated collection pit. The following diagram displays the setup of a mancala board:

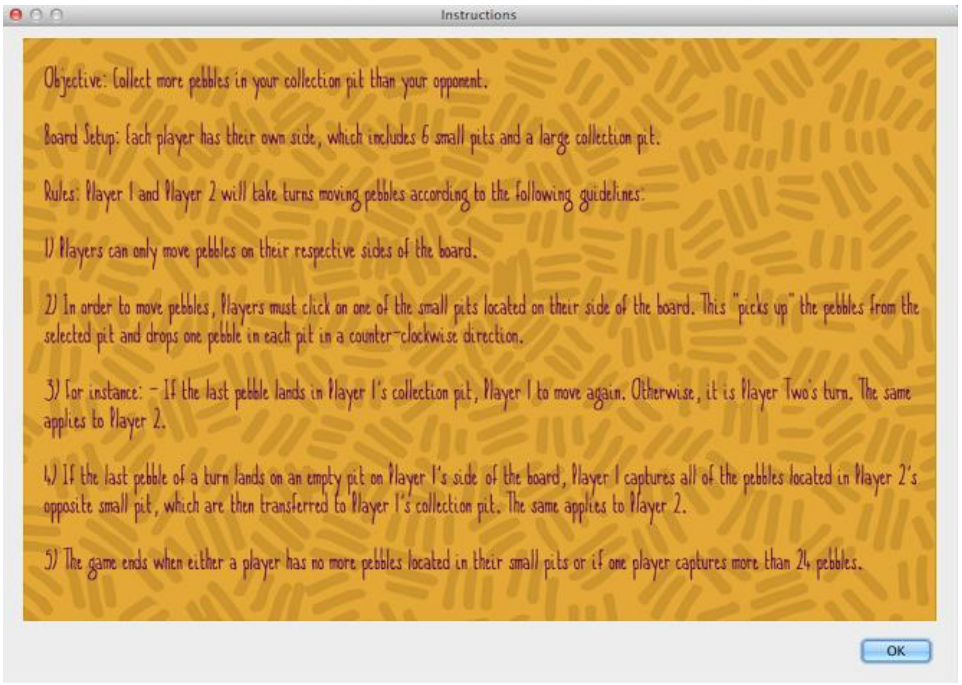



Each player has a side consisting of 6 small pits as well as one large collection pit. At the start of a game, each player has 0 pebbles in their collection pits while the 12 small pits on the board contain an initial count of 4 pebbles. Player 1 and Player 2 take turns moving pebbles according to the following guidelines: A player may only move pebbles from a pit located on their respective side. A player picks up the pebbles, and drops one pebble at a time into pits in a counter-clockwise direction. A player may not drop a pebble, however, into their opponent's collection pit. If a player drops their last pebble into their own collection pit, they may move again. Otherwise, it is the opponent's turn. If a player drops their last pebble into an empty pit on their side of the board, they may capture this single pebble along with the pebbles located in their opponent's opposite small pit, and place these pebbles into their designated collection pit. The game ends when either one player runs out of pebbles on their side of the mancala board or if one player captures more than 24 pebbles.

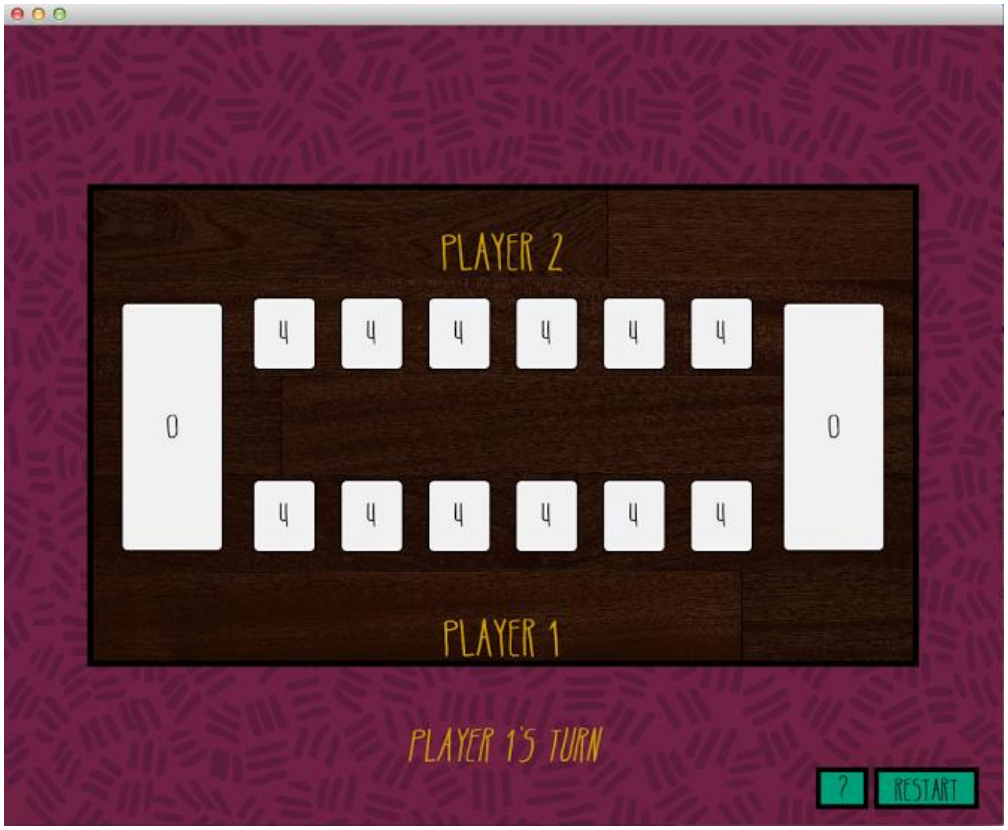
*Running Program*

In order to run the program, open and compile the *MancalaGUI.java* file. The following window should pop up on the user's screen:


This start menu contains 2 buttons. If the *HOW TO PLAY* button is selected, a separate message dialog box containing game instructions pops up. Once the user reads the instructions, they may select the *OK* button which closes out of the dialog box, returning to the start menu.

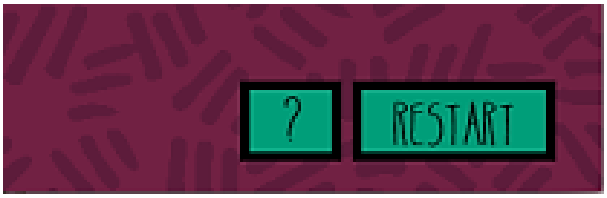


In order to play, the user must select the *START* button which redirects to a new panel containing the game board as is shown below:

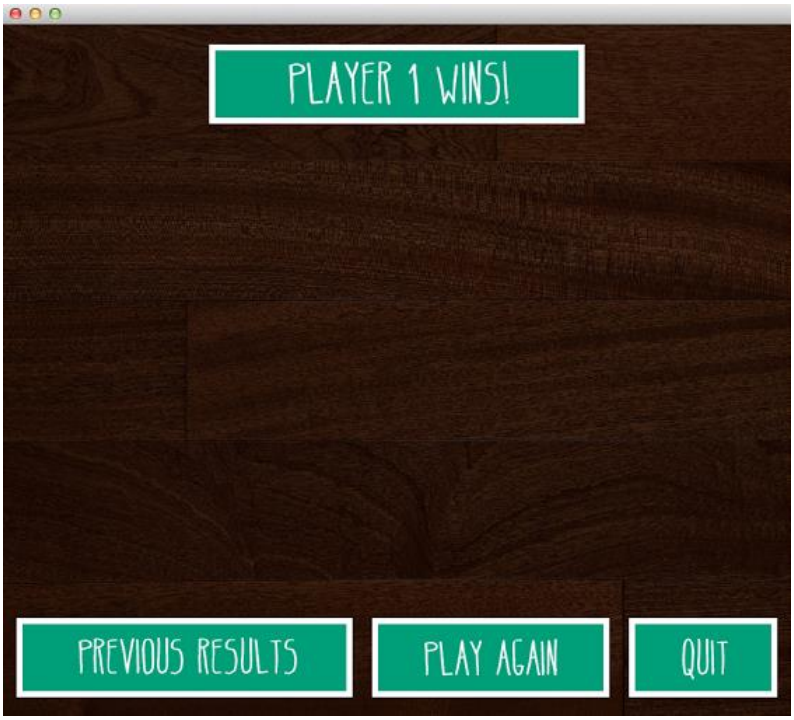



Two human players are required in order to play this version of Mancala. To start, player 1 must click on one of the six small pits located on the bottom half of the board, labeled *PLAYER 1*. As soon as a pit is clicked, the entire board updates to reflect changes in pits' counts as pebbles are "picked up" and "moved" along the board. If player 1 lands their last pebble into their last pit, a label on the bottom half of the screen will prompt player 1 to take another turn. The program is designed in such a way that no pebbles are moved if player 1 clicks on a collection pit and/or player 2's small pits. Back and forth, player 1 and player 2 continue to take turns as prompted by labels located on the screen until the game is over.

During any point in the game, the user has the option to restart the game by selecting the *RESTART* button. Additionally, if they would like to reread the instructions, they may select the "*?*" button, which causes the instructions dialog box to pop on the screen. Pressing the "*?*" button does not restart/interfere with a game currently in progress, meaning a player can resume the game after re-familiarizing themselves with the instructions.

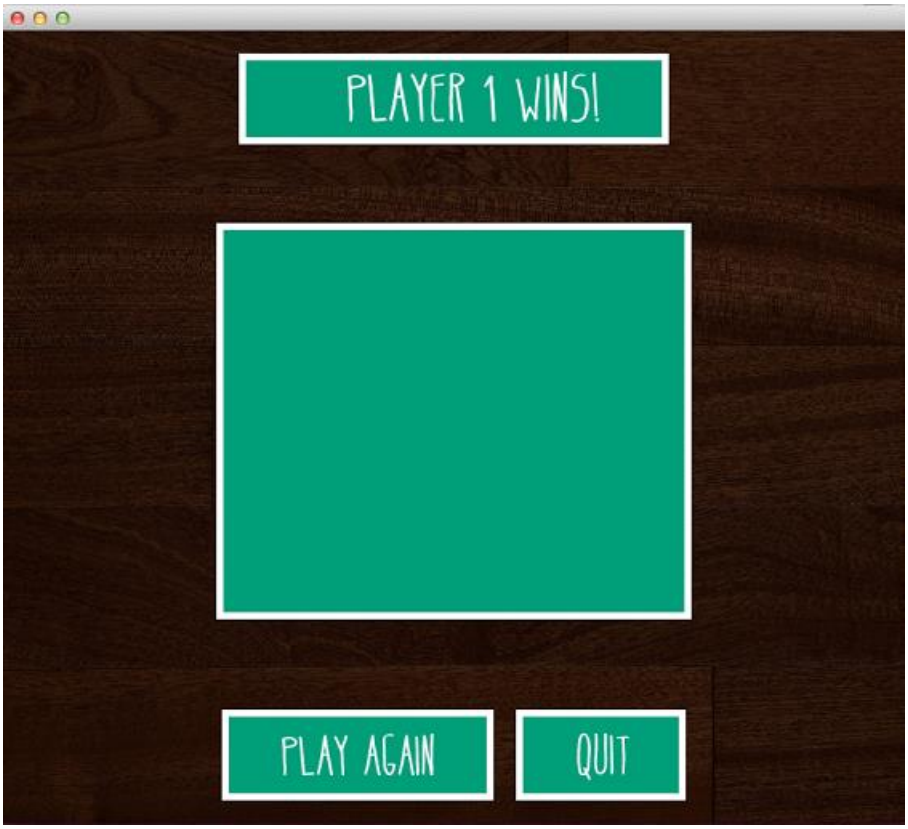


When the game is over, a frame pops up indicating the result of a game (win or tie). A user has the option to select one of three buttons: *PREVIOUS RESULTS*, *PLAY AGAIN*, or *QUIT*.




When the user selects *PREVIOUS RESULTS*, a box displays the winners of previous rounds. Note: the program only tracks the winners of previous rounds while the program is running. If at any point, the user closes out of the program, the stack storing these previous winners is reset.

Due to space constraints, the program only stores the previous winners of up to 4 rounds. If a user decides to play a fifth round without closing out of the program, the *PREVIOUS RESULTS* button is disabled, and previous round results are not displayed. The button will only reappear if the user restarts the program.




Selecting the *PLAY AGAIN* button causes the pop-up result frame to close, returning to the GUI game screen. A new game is instantiated. If a user selects the *QUIT* button, however, the program is closed.

Font does not properly display on a PC. Please run the program on a Mac. Enjoy!

# Technical Report:

- This program consists of the following classes:

  *GAME COMPONENT:*

  1) Pit.java
     *A parent class that creates generic pit objects*.

  2) SmallPit.java
     *Extends Pit.java. Represents the small pits on Mancala board.*

  3) Player.java
     *Creates a player*

  4) MancalaBoard.java
     *Creates a LinkedList representation of the Mancala board.*

  5) LetsPlayMancala.java
     *Creates a new playable game of Mancala.*

  *GUI COMPONENT:*

  6) BkImagePanel.java
     *Creates a panel with a read-in background image.*

  7) StartFrame.java
     *Creates a frame with a panel containing the startup menu/splash page.*

  8) MancalaPanel.java
     *Creates a panel that runs the Mancala game.*

  9) MancalaGUI.java
     *Runs the GUI.*

- ADTS

    1) LinkedList (MancalaBoard.java)

        *We decided to use a LinkedList to create the game board. The LinkedList stored all of the pits in both a specific order and indexed position. Because the pits needed to contain references to one another so pebbles may be passed from one pit to the next, a LinkedList was a more appropriate ADT to use than an array.*

    2) Stack (MancalaPanel.java)

        *A stack was used in order to store the winners of previous rounds. We decided it would be more appropriate to use a stack since we could easily pop the most recent winners from the top of the stack.*