Mancala Game using AI

Team members:

Ahmed Khaled Mohamed Gamal ID: 1600079
Ahmed Sameh Mostafa Abdelhamed ID: 1600085
Ahmed Hossam Kamal Mahmoud ID: 1600057
Ahmed Yehia Ahmed Mohamed ID: 1600219
Mohamed Abdallah Mostafa ID: 1601222

1-Brief Description:

Game description:

We are implementing Mancala game,We are implementing Mancala game, It is a board game contains 6 slots + Mancala slot for each player, each slot has 4 stones and both Mancalas are initially empty, first pick which player to play first, a play chooses a slot to spread its stones anti-clockwise to the neighbor slots if passed by the player Mancala drop a stone, if passed by an opponent Mancala skip it, if the last stone is in your Mancala have another turn, if a player run out of stones in his slots all stones in the opponent slots goes to the opponent Mancala and the game ends, the player with max number of stones in Mancala wins, this game has two modes without stealing and that what i described above, and with stealing mode, in stealing mode you steal stones from the opponent if your last stone is at your slot and this slot is empty you take all the stones in the corresponding opponent slot

Implementation description:

the board is created as 1D array, from the first slot to 6th slot are holes and 7th is Mancala of the AI player, from 8th slot to 13th are the holes and 0 is the Mancala of the human player, either the AI or a human player chooses a slot or a hole to spread stones in it is done by shift() function that supports stealing or non-stealing option, modify the board after each move, if the last stone drops in the player's Mancala he/she gets another turn, the AI is implemented by MiniMax with using Alpha-Beta pruning, to get the best play, this is done by a heuristic
error handling:
1-if the user enter a slot that have no stones, he/she is notified to try again
2-if the user load a game that does not exist, he/she is notified that there is no previous game and a new game will start
3-any wrong input the program will ask the user to input a correct expected value

Bonus:

1- save and load the game
2-easy/medium/hard levels option based on depth level

2-Functions Description:

1- **print_grid(board):**
input params: board (1D array represents the game board)
output: print the board

2-**print_rules():**
print the available slot numbers the user can use to pick a slot and its place on the board

3-**shift_the_stones(slot_number,player,board,steal=True):**
the function spread the stones of the chosen slot and determine if the player has another turn or not
input params: 1- slot number: contain stones to be spread
2- player: flag to determine which player has the turn
3-board: the current board state of the Mancala game
4-steal: Boolean value to use stealing option or not
return another_turn,virtual_board: another turn if the player got another turn,
virtual_board: the new board after any player's move

4- **check_does_player_finish(board):**
check if the last player's move end the game or not
input params: board
returns True of False, True if the game ended, False if not

5-**def end_board(player,board):**
end the game if any player's slots are all empty by removing all the stones on the opponent side to his Mancala
input params: player, board

6-**all_possibles(board,player,steal):**
get all possible states after the current state of the board of the current player's turn
input params: board, player,steal
returns possible_boards, a list of lists where the inner lists contain two elements, a board after a possible move and boolean value determine if the player has another turn or not

7-**def settings():**
function to ask user about the game set up, difficulty, choosing whom to play first, load the game or start a new game, use stealing or not

8-**def minimax(board, alpha, beta, depth, max_depth, next_step ,steal):**
A recursive function that apply Alpha-Beta pruning algorithm, the base case of that function is that if the depth of the tree from the root reaches the maximum depth the evaluation function is used and heuristic of the current state is calculated, after that the function uses "all_possibles" function to get all possible next states and loop over them, on each iteration it call itself recursively with the next state board and incremented depth, then it updates alpha or beta "depending on whether it is maximizer or minimizer" and finally return new computeed alpha and beta and the selected index

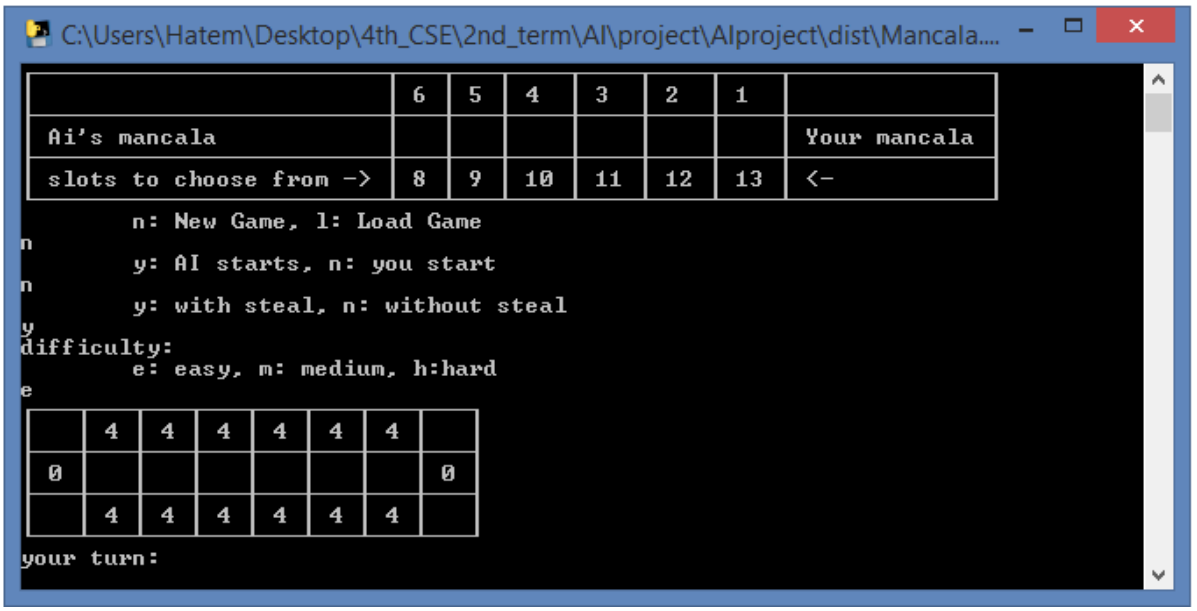
Input params: board, alpha, beta, depth, max_depth, next_step,steal
Return: newAlpha,newBeta,nextStateIndex

9-**def evaluate(board):**
function returns heuristic to be used in Minimax function,
the used heuristic: sum(all stones in the slots and mancala for some state at the AI side) - sum(all stones in the slots and mancala for some state at the human player side)
Return: Heuristic

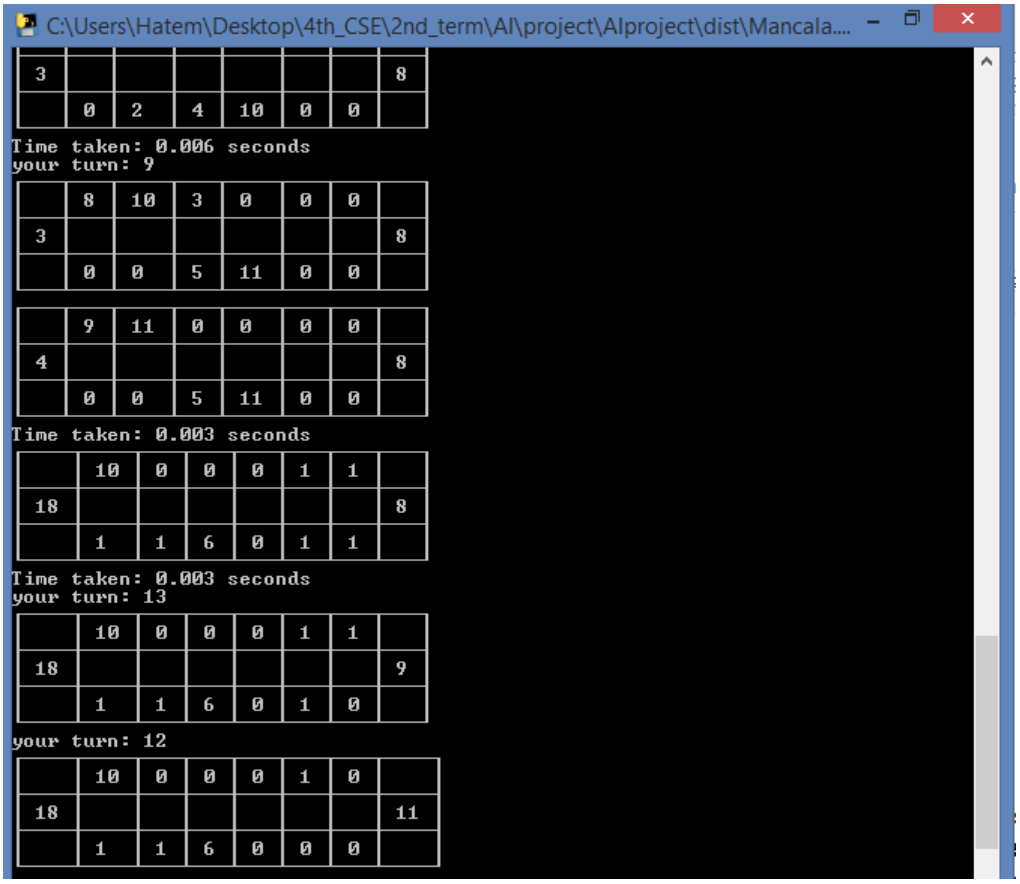
10-**indexToSlot(board,newboard):**
The function converts the index of the best selected next state to slot number that can be used by the board to perform that move
input params: board, newboard
Return: slotNum

3-User guide

1-After running the exe file, you first need to input:
n: to start a new game or l: to load a previous saved game
2-you have the slot values to choose from during your turn,
3-you pick either to have the first or the second move
4-you choose the difficulty of the game



Here is a game sample


C:\Users\Hatem\Desktop\4th_CSE\2nd_term\AI\project\AIproject\dist\Mancala....
3 8
0 2 4 10 0 0
Time taken: 0.006 seconds
your turn: 9
8 10 3 0 0 0
3 8
0 0 5 11 0 0
9 11 0 0 0 0
4 8
0 0 5 11 0 0
Time taken: 0.003 seconds
10 0 0 0 1 1
18 8
1 1 6 0 1 1
Time taken: 0.003 seconds
your turn: 13
10 0 0 0 1 1
18 9
1 1 6 0 1 0
your turn: 12
10 0 0 0 1 0
18 11
1 1 6 0 0 0


C:\Users\Hatem\Desktop\4th_CSE\2nd_term\AI\project\AIproject\dist\Mancala....
1 3 7 0 1 1
Time taken: 0.003 seconds
your turn: 13
0 0 0 0 1 1
22 12
1 3 7 0 1 0
your turn: 12
0 0 0 0 1 0
22 14
1 3 7 0 0 0
0 0 0 0 0 0
23 14
1 3 7 0 0 0
YOU WIN
0 0 0 0 0 0
23 25
0 0 0 0 0 0
Type Any Key to quit

4-Roles:

Ahmed Khaled Mohamed Gamal : Settings Function(including saving and loading the game) end_board Function - report
Ahmed Sameh Mostafa Abdelhamed : Evaluate Function - indexToSlot Function - main loop
Ahmed Hossam Kamal Mahmoud : shift_the_stones - check_does_player_finish - all_possibles Functions
Ahmed Yehia Ahmed Mohamed : minimax Function and executable file
Mohamed Abdallah Mostafa : print_grid - print_rules Functions