Project Report: Artificial Intelligence

Supervised by: Dr: Manal Morad

Team members	ID
Fatma Emad Elden Ibrahim	1600973
Fatma Elzahraa Mohamed	1600968
Asmaa Ramdan Abdelbaset	1600268
Aya Ahmed Ismail Mohamed	1600361
Eman Mohamed Abdelmohsen	1600354

Game Description:

Setup:

The game is played by 2 players, The game board is made of 2 rows of 6 small holes known as “pockets”, and large holes on opposite ends called “mancalas” or “stores”.

Each player has 6 pockets directly in front of him and a store to his right, Place 4 stones in each of the 12 pockets.

Objective:

The object of this game is to see which player can get the most stones into his store.

Gameplay:

One player will start the game by picking any pocket containing stones from his own side.

The player will remove all the stones from that pocket, and deposit one stone at a time into neighboring pockets going counterclockwise until the stones run out.

If a player encounters his own store, a stone is deposited in it, but if he encounters the other player’s store, that store is skipped over.

If the last stone is deposited in the player’s own store, the player gets another turn.

Stealing mode:

If the last stone is placed in an empty pocket on the player’s own side, the player takes this stone as well as the other player’s stones across from the empty pocket landed in, and places them in his own store.

The Algorithm:

We implement an algorithm to play against the human player, this algorithm is based on minimax algorithm with alpha_beta pruning to choose the best move for the algorithm in each state, the algorithm is based on Depth first search

The implemented game has the following features:

- The user can choose to play first or make algorithm to play first
- The user can choose to play with stealing mode or without stealing mode
- The user can choose which level of difficulty he wants to play, there are 3 levels (easy level, medium level, hard level)

The utility functions used by the algorithm:

game_ends(Ai_side,Player_side):

This function takes the whole board (Ai side, Player side) to check if game ended or not , and if the game ended then It collects the stones of each player (human player & AI algorithm) in his store and returns true

Final_score(Ai_side,Player_side):

It is evaluation function used in the max_min algorithm when the game ends during the search algorithm trial

experimental_score(Ai_side, Player_side):

It is evaluation function used in the max_min algorithm at leaf node (it is difference between two player's scores)

minimax_alphabeta(player_layout,AI_layout, alpha, beta, depth, is_Max,withstealing):

It is the function of minimax with alpha_beta pruning. We call this function when AI player wants to find the best move from all possible moves , This function is called recursively to find all possible moves of (AI player & human player)with the given depth until it reaches to the leaf nodes then it calls evaluation function at this node and it performs alpha_beta pruning to find the best move and pruning helps to make search faster , finally it returns the index that represents the best move(higher score) for AI player , depth is the parameter that controls the difficulty of game level , as depth increases as the difficulty of game level increases and time consuming in search increases

make_player_move() & make_ai_move():

these two functions make player and algorithm play according to the game rules,

make_player_move() is related to human player and make_ai_move() is related to AI player , each one of them check the following rules:

- The player should take another turn if the last stone is placed in his store
- Each the player can put stones in all the pockets and his store but he should skip the opponent store
- if the game in the stealing mode, then performs stealing

player_move(player_layout):

This function take the number that represents the move from the human player, and check if the move from his side and check if the user choose an empty pocket, then it calls function make_player_move()

ai_move(ai_layout):

In this function we call minimax_alphabeta() function to get the best move for the algorithm , then it check if the game ends , then it calls make_ai_move() function to make AI player plays with this move

layout(): print the game

board

playing_mode():

This function asks the user:

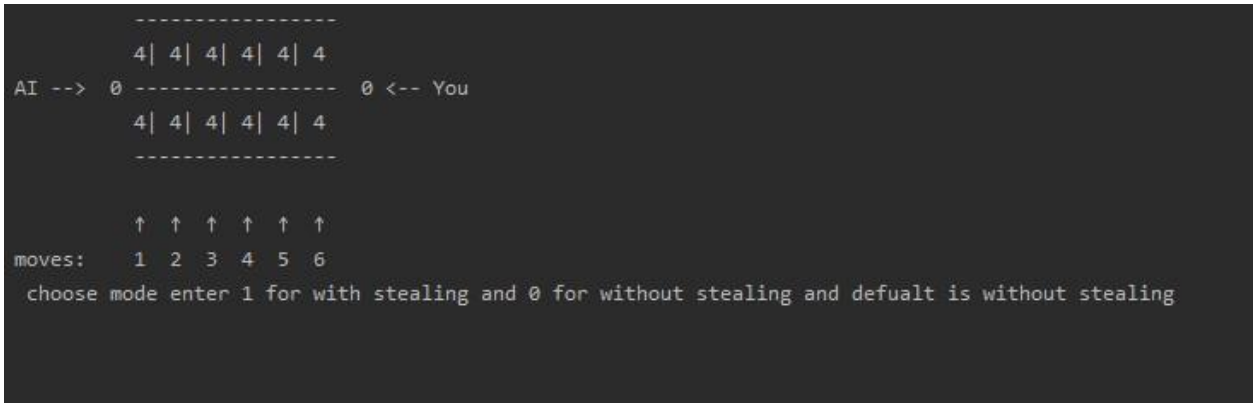
- if he wants to play first or make the algorithm play first
- to choose game with stealing mode or without stealing mode
- which game level he wants (easy level, medium level , hard level)

Then it call ai_move() & player_move() functions and checks if the game ends

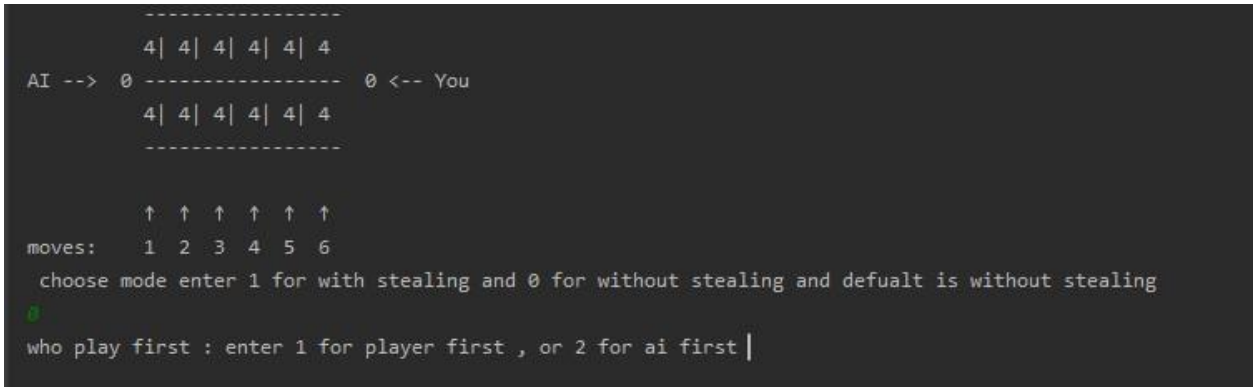
User Guide:

When the game starts the initial layout of the board will be printed then the program will ask for the following information:

- choose mode enter 1 for with stealing and 0 for without stealing and default is without stealing



- who play first: enter 1 to play first, or 2 to make Ai play first



- finally choose the level of the game:
  - Enter 1 for easy mode
  - Enter 2 for medium mode
  - Enter 3 for hard mode

```
         ----------------
         4| 4| 4| 4| 4| 4
AI -->  0 ----------------  0 <-- You
         4| 4| 4| 4| 4| 4
         ----------------

         ↑  ↑  ↑  ↑  ↑  ↑
moves:   1  2  3  4  5  6
 choose mode enter 1 for with stealing and 0 for without stealing and defualt is without stealing
0
who play first : enter 1 for player first , or 2 for ai first 1
enter 1 or 2 or 3  : 1-easy_mode ,2-medium_mode ,3-hard_mode 1
```

- The game will start at player turn enter the number of pocket in which you take the stones from then the board layout will be updated and the AI will play.

```
Player turn: 1
            ------------------
            4| 4| 4| 4| 4| 4
AI -->  0 ------------------  0 <-- You
            0| 5| 5| 5| 5| 4
            ------------------

            ↑  ↑  ↑  ↑  ↑  ↑
moves:      1  2  3  4  5  6
AI move ---> 4
            ------------------
            5| 5| 5| 0| 4| 4
AI -->  1 ------------------  0 <-- You
            0| 5| 5| 5| 5| 4
            ------------------

            ↑  ↑  ↑  ↑  ↑  ↑
moves:      1  2  3  4  5  6
AI move ---> 3
            ------------------
            6| 6| 0| 0| 4| 4
AI -->  2 ------------------  0 <-- You
            1| 6| 5| 5| 5| 4
            ------------------

            ↑  ↑  ↑  ↑  ↑  ↑
moves:      1  2  3  4  5  6
Player turn:
```

- The last point will be repeated until the game ends.

The Members' work:

The project splits to the following functions:

Member	functions
Fatma Emad	Implements make_player_move() function Implements game_ends() function
Fatma Elzahraa	Implements make_ai _move() function Implements final_score() function Contributes in minimax_alphabeta() function
Asmaa Ramdan	Implements minimax_alphabeta() function Implements experimental_score( )function Made the video
Aya Ahmed	Implements ai _move() function Implements playing_mode() function Contributes in make_ai _move() function Made the report
Eman Mohamed	Implements player_move() function Implements layout() function Contributes in make_player _move() function