**Nayan Halma:**

#initializing our main class to play game  
#Parameters:  
#Board size:8,10,16  
#Timelimit : in seconds  
#player1="RED"or "GREEN"  
#player2="RED or GREEN"  
#AI="AI" if you want Bot to play as player 2 else None for Human  
#AlphaBeta= "ON" or "OFF"  
# ALways the second player is assigned as AI  
game = HalmaGame(8, 10, "RED", "GREEN","AI","ON") #For Human vs AI  
# game= HalmaGame(8,10,"RED","GREEN",None,"ON") # for human vs Human

class HalmaGame: This is the main class of the game. It assign players and it is responsible to create the board internally to perform the calculations such as moves from one tile to other.

Play():play function runs a loop until the game is not finished .. each player gets one chance to play one move if a user forget to play his move within a time limit then he cannot make his move until his next turn.

Endgame(): This function checks which player won the game. And responsible to terminate the game.

Halmaguimode class: Gui Class which is made using tkinter. this class handels all the functionality of GUI to show the moves,Show whos turn is currently now.It also shows all the information if AI is playing that is from which tile to wich tile ai has moved.

Timeren() This function is the timer of our game which is initialised evertime when a player get his turn. User can change the timer limit at the start of the game.

Makegrid() This function is used to create grid for visualization and calculation purpose.

Makechar() This function creates the tokens for each player.

Onpress() Function works on clicking a token of a game.if a player press one of his pice of token this function shows all the possible moves in the interface with a boundary Highlighted  
And if a player presses another tile to move.  
this function is responsible to make a move using interface.

Game File

Game Class : Our Game Class which handels and Genrate moves in the back.It is also responsible to generate Jumps while making moves.

Checkjump()Function to Check Jump on the board based On the current Positon of a player  
jumps can be made if other player is in neighbouring tile and the next tile to the other player is Empty

Validatemove() this function checks that if a player is making a move then it is validated that player is making a correct move or not.

char\_movement() This is the Function that is used to move the token from one tile to other tile. Its response is recorded and referred in the Gui

distance\_calculation() This function calculates the distance from the current node to the goal node. This is used when AI is playing so that ai can calculate all the possible moves and retuns the best move.

Calculategoal() This function calculates all the distance form node to the all by calling the distance calculate function internaly for every valid token.

CharTurnCord() This function takes all the valid moves in input and returns the coordinates of each move.

Minimax() This is our main algorithm. By using this algorithm our AI plays the game by calculating the best move. AI explore all the moves and retuns the best move. It also uses alpha beta purning inside which can help in reducing a lot of time.

Player\_Char\_moves() This function returns all the possible moves for a player. This function is used by AI to get all the possible moves.

Char\_Move\_Temp() This function temperory move the token internally just for calculation it is used by ai to find the best possible moves.so it calculates all the possible moves by moving a token.

AiPlay() This function handels all the functionality of the AI from start to end. It calls all the functions that is used by AI to play a move. Once best move is returned it plays that move. And return the status of movement.

CharacterUtil Class.

CharacterUtil Class: This class is used to assign players a specific color, home and goal coordinates.

Starposchar() This function gets all the starting position of the Players i.e goal , home coordinates.

all\_char\_opp() This function checks that a game is finished or not. Game is finished when all the tokens of player 1 reach the goal. That is home coordinate of player 2

charlocCheck() This function check the current location of the token.

charHomeCheck() Function to check the home cordinates

charGoalCheck() Function to check the goal coordinates

---------------------------------------------------------------------------------------------------------------------------

Brain phase:

As in phase 1 we created a interface and all the functionalities for halma game so that Two players can play the game. We created gridded interface and some small functionalities which returns the important information such as valid moves and jump. Now in brain phase we created a brain for halma so that a player can play the game with AI. By using all the previously built functions its become a easy task to create a brain because we have mostly everything that we needed to make a brain for AI.

As we created a gridded interface we have the coordinates of each and every tile so we added some more functionalities like Calculating the distance from tile to the goal by using the distance formula. This is used by AI to calculate the distance for its piece/token form a tile to the goal tile.

We created Minimax algorithm that is the main brain of our AI player. We used minimax as it’s a kind of backtracking algorithm that finds the optimal move in games such as Chess, Halma. It has a maximizer and a minimizer. Maximizer try to get the highest score and minimizer try to get the lowest score. Using the minimax algorithm our AI generates the best optimal move over all the possible valid moves. Once AI get the best move it returns the move and a our AI plays that move within a time limit.

Once we implemented minimax we added Alpha beta purning which is a optimization for our minimax algorithm. It help in reducing the calculation saves time and get the moves as fast as possible.

Alphabeta purning shows us very promising results as we tested when making first move when alpha beta is off AI will take about 6 seconds to make the move.

And when we tuned on Alpha beta purning AI takes about 2 second to make the same move.

When player 1 is making a move then player 1 has some timelimit to make a move. As player 1 done with a move. Then our algorithm shifts the turn to the AI if ai is playing. Then AI gets all the possiblemoves using minimax in a recursive way gets the best move and make a move. Once ai done with the move its status is shown in a bar that we added newly.

This new bar that we added shows the status of Ai such as from which tile to which tile AI played the game.

Whoever wins the game. Player name is displayed as winner and game closes automatically.

NOTE: When alpha beta purning tuned off AI takes 6 seconds to make first move. And when Alpha beta is turned on AI will take about 2 seconds to make the same move.