Assignment 2

Problem 1.

The problem is to find the possible moves that don't make the current player lose the game immediately and selecting a move which makes the other player's future moves more restrictive. Passing on to the other player a board which is harder for him/her to win. The problem also includes the possibility to play more aggressively if the other player is already in a bad state.

State space: Is the possible configuration of the n-k-cho-cho game. Where each player can place each respective black or white marble as per their choice on n by n board by following a legal move. A legal move is defined as when a respective player can place a either a white or black marble on the board where there is an empty space

Start state: The start state is an empty board of n by n, consisting of the size (n) specified by the user. It can also be assumed that white player will always make the first move. The board also includes a parameter 'k' which is the max quantity in which a marble of a particular color can be placed in a row, column or diagonal.

Goal state: There are 3 specific goal state

1) **Player 1 wins (white marbles) – when player 2 losses i.e player 2** completes a row, column or diagonal of size k with black marbles, where k is provided by the user.
2) **Player 2 wins (black marbles) – when player 1 losses i.e player 1** completes a row, column or diagonal of size k with white marbles, where k is provided by the user.
3) **Draw –** when there are no more blank space on the board where either of the player can place any marble and none of the players have placed their respective marbles in k quantity in any row, column or diagonal

Successor function: The successor function take the current board looks at all available place (blank places) places a marble in a legal move fashion with reference to the player and returns the board. These returned board are further evaluated by another function based on the heuristics/evaluation and finally a suggestion is made to the user.

Cost: The cost is uniform as the we can make only one move in each successor and the cost of making the move is constant.

Heuristic function: The heuristic function is composed of two parts

- Avoid immediate loss idea takes the board and calculates the place where the current player should not make a move or place the marble in order to not lose immediately- This part is based on the condition that the current move to be suggested does not meet the terminal or goal state with the current "k"

- Future evaluator of the all available moves after the immediate loss function this parts calculates the move which makes the next moves of the other player not so lucrative.

Assumptions: The white marble player is the user for us and we are recommending the moves for him/her

Parts of the program:

The program consists of the following parts:

For Class Board we have the following functions .

- **add_piece** - adds a piece on the board
- **get_available_place**- returns the number of empty places that are available to make a move
- **get_successors**- returns the new intermediate board along with the player and some evaluation metric to choose the next best move
- **has_lost**- check whether the terminal condition has met or not
- **get_diagonals-** returns the diagonal elements based on the configuration of the board which is the result of "n' the size of the board

For Class ChoCho we have

- **Attributes –** state, whites and blacks array, white and black distribution centroids and alpha and beta values which are weights for evaluation
- **print_stuff_single_line**- print the board in the required format

Explanation of the program:

- The program initially takes the input form the user
- Generates the successors and passes each successor state to the evaluation part
- Check the place which the player1 or 'W' should not play in order not to lose the game immediately and make them unavailable to play for the white marble placement
- Then the program check all the possible move in a given time frame to see which move reduces the free available places for the next player and makes that move
- Finally based on the size of the board 'n' and 'k' there is a possibility that the program will apply min max algorithm to ensure at-least a draw if possible from the given configuration of the board