**COURSE: ARTIFICIAL INTELLIGENCE**

**ASSIGNMENT 1: Checkers Games**

**AVANTIKA SHARMA: 1910110104**

**DIYA SACHDEV: 1910110140**

**KHUSHI CHAWLA:1910110203**

**Language Used :** Python

**Package used :** Pygame

**GUI :**

Pygame GUI is a module to help you make graphical user interfaces for games written in pygame.This module offers control over the pygame display. Pygame has a single Surface that is either contained in a window or runs full screen. Once you create the display you treat it as a regular Surface.When the display mode is set, several events are placed on the pygame event queue.

Pygame will register all events from the user into an event queue which can be received with the code **pygame.** **event**. get() . Every element in this queue is an Event object and they'll all have the attribute type , which is an integer representing what kind of event it is. Pygame.quit() is returned once we have a winner or if move count exceeds 50.

**Class**

1. **Constants -** which set default values for height, width of checkers board, colors etc.
2. **Class Checkers\_Game** - handles which piece is selected, whose turn out of comp / human is it and where all we can move the piece
3. **Class Checkers\_Token**- represent a checkers piece on the board - drawing a piece and moving places
4. **Class Checkers\_Board** - represents the checkers board - drawing the board and all its pieces on specific positions and moving / capturing positions.:

**Functions**

1. **game\_state\_update( self ) -** updates the game state
2. **declare\_winner( self ) -** declares winner of the game
3. **move\_selection( self, row , col ) -** on first call, it finds all the moves and then on second  call moves to that position.
4. **token\_move( self, piece, row, col ) -** moves the previously selected piece to the given row and col
5. **Draw ( self, window ) -** draws a piece on screen on respective position
6. **box\_checkers( self, window ) -** draw all the edge lengths on the window
7. **utility\_mark( self ) -** calculate utility value at cut-off. It gives priority to killing rather than making king which makes it perform better
8. **token\_list( self, color )** - returns all pieces of given color
9. **Move ( self, piece, new\_row, new\_col )** - moves a piece on the new position on boards and updates board and piece accordingly
10. **Remove ( self, pieces ) -** removes list of pieces from the board
11. **move\_list( self, piece ) -** returns all possible pieces as a dictionary
12. **leftwards( self, beg, end, inc, color, l, removal = [] ) -** moves pieces in left diagonal
13. **rightwards ( self, beg, end, inc, color, r, removal = [] ) -** moves pieces in right diagonal

**Random Functions and Algorithms:**

The Random Algorithm follows the following 3 steps :

1. It checks whose turn it is - computer or human. Incase of human it lets the human does whatever he/she wants to
2. For computer - It makes a list of possible solutions
3. Out of the possibilities, it takes random int i in range of list and takes that move.

**Mini-Max Functions and Algorithms:**

The name "minimax" comes from *mini*mizing the loss involved when the opponent selects the strategy that gives *max*imum loss, and is useful in analyzing the first player's decisions both when the players move sequentially and when the players move simultaneously.

The Mini-Max Algorithm follows the following 3 steps :

1. if depth is 3 or there is no winner the return updated board and updated score
2. If turn =1 then max = -INF and finds out possible moves and saves it in a list. Recursively calls minimax by increasing the depth and switching the turn and runs iteration of the list if max<val then save and return best possible move
3. Else min = INF and finds out possible moves and saves it in a list. Recursively calls minimax by increasing the depth without switching the turn and runs an iteration of the list if min>val then saves and returns the best possible move.

**Contribution:**

1. **Avantika(1910110104) - 33%**
2. **Diya(1910110140) - 33%**
3. **Khushi(1910110203)- 33%**