CSE537 Artificial Intelligence
Project 2 – Game Search (Connect Four)

Team Details

Name	**SBU ID**	**Net ID**
Shahzeb Nihalahmed Patel	110369918	shahpatel
Rahool Paliwal	110570213	rpaliwal

You must execute the file lab3.py

1.Minimax search (20 Points):
We have implemented minimax algorithm for ConnecFour game, in file basicplayer.py. Refer line 90 for signature of the function. This is adapted from
https://en.wikipedia.org/wiki/Minimax

To run basic player vs human player, uncomment lines 16, 17, 18 in lab3.py. The rungame has implementation to find time. We have a static variable for each algorithm which keeps a track of expanded nodes. We have printed it under the rungame call to print the expanded node.

Player1 is basic player
Player2 is human player

Player 1 (☺) puts a token in column 0
Win for ☺!

```
 0  1 2 3 4 5 6
0    ●
1 ☺  ●
2 ☺  ☺
3 ☺ ● ● ☺
4 ☺ ● ☺ ☺
5 ● ● ● ☺
```

('Time required to run : ', 9.001305)
('Nodes expanded :', 22399)

Q2. Better Evaluation Function

The new evaluation function is implemented from line 36 in basicplayer.py

This is the new evaluation function, for every column, it will find out the largest possible sequence for the current node in all directions. It will return the length of longest number of consequence nodes in all directions. We do this for every column. If the top node is current player's node, it is added to current player's score, else it is added to the other player's score

Uncomment line 23, 24, 25 in file lab3.py

Running new player against basic player
Player 1 = new Player
Player 2 = basic player

Output
Player 1 (☺) puts a token in column 5
Win for ☺!

```
   0  1  2  3  4  5  6
0  ●  ●  ☺  ●  ●
1  ☺  ●  ●  ☺  ☺
2  ●  ●  ☺  ●  ●
3  ☺  ☺  ●  ☺  ☺
4  ●  ●  ☺  ☺  ●
5  ☺  ●  ☺  ☺  ☺  ☺
```

Q3. Alphabeta Search (30 Marks)

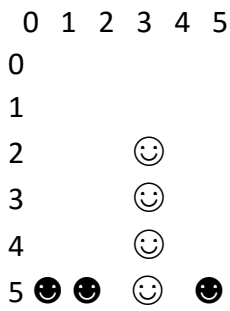
We have implemented alpha beta pruning on minimax algorithm in file lab3.py from line 67.
https://en.wikipedia.org/wiki/Alpha%E2%80%93beta_pruning#Pseudocode

Uncomment line 169 and 170 in file lab3.py to run alphabetaplayer against random player

Player 1 = alphabeta player
Player 2 = random player

Output:

Player 1 (☺) puts a token in column 3
Win for ☺!



('Time required to run : ', 1.270058)
('Nodes Expanded :', 3175)

Note the number of expanded nodes are less as compared to what we got in minmax solution.

4. Generalization of game

a. Connect K game (10 Marks)

We have implemented kth method by defining a new member variable as win_k in ConnectFourBoard class.
As per the problem set specification we have set as,
run_game(alphabeta_player, human_player, ConnectFourBoard(k)) with default value of k as 4.
Refer line 472 in line connectfour.py file and the constructor of ConnectFourBoard class for implementation details.

The default value is set to 4.
To run connectk write, the following line in lab3.py
run_game(alphabeta_player, human_player, ConnectFourBoard(k = 3)) to have connect 3 game.

5. Documentation + Report.

Every function implementation has comments describing the approach and substantial steps in the implementation. Also each output will print the time required to execute the game and number of nodes expanded. (In case of running a player against human player, the time includes the delay caused by human).