AI Assignment – 2
Aaryan Chauhan – 2021A7PS2595G

(a) and (b) and (d): (c done at end)
In these sections we look at performance of bot under different evaluation functions and for different depths (3 and 5).
Metrics: Number of times won out of 50, and average moves taken to win.

EVALUATION FUNCTIONS:

(i) Most simple evaluation function used was counting game tree player's pieces (score) occuring in same rows, columns and diagonals and subtracting the opponent's "score" then returning it, without attaching any kind of weights to each combination.

(ii) Next, gave linear weights to each kind of combination (4 or more in a row got weight – 4, 3 in a row – 3, 2 in a row – 2, 1 individual – 1).

(iii) Next, gave exponential weights (4 or more in a row – 100000, 3 in a row – 1000, 2 in a row – 100, 1 individual – 10).

(iv) Tried another evaluation function which prioritised center positions and gives more weight to more number of player's two-coin configurations, but (iii) is most suitable for our needs.

PLEASE NOTE:

Here I have combined all the statistics of part (a), (b) and (d). The thing to note here is that evaluation functions (i) and (ii) are poor. So, upon increasing the depth, it may give more importance to worse actions. Thus here the data reflects this.

Evaluation function (iii) is optimal for our needs, and we even see that we win almost all of the games upon increasing the cutoff depth to 5 and the average moves to win have reduced significantly especially compared to (i) and (ii).

TESTBENCH: Amended the PlayGame() function for win rate, average moves performance.

TESTCASE: is passing in 3 moves using the evaluation function (iii).

Implemented alpha-beta pruning as well, but since it only affects time taken by the algorithm and not wins, didn't include it in the table.

Statistics:

Evaluation Function (i):

Depth	Average moves to win	Average win rate (50 games)
3	29.66	29
3	27.71	28
3	29.08	28
Average	28.82	28.33
5	26.96	25
5	25.42	21
5	27.28	25
Average	26.55	23.67

Evaluation Function (ii):

Depth	Average moves to win	Average win rate (50 games)
3	28.31	32
3	26.89	34
3	28.17	35
Average	27.79	33.67
5	27.18	27
5	28.07	26
5	26.67	21
Average	27.31	24.67

Evaluation Function (iii):

Iteration	Average moves to win	Average win rate (50 games)
3	20.71	42
3	22.35	40
3	19.30	43
Average	20.78	41.67
5	19.16	45
5	22.13	47
5	20.31	45
Average	20.53	45.67

We can clearly observe: (iii) > (ii) > (i)

Didn’t consider (iv) as (iii) worked optimally

(c) Move ordering heuristic

Using a simple move ordering such that the bot prefers putting a coin somewhere in the center than the corner (i.e. most weight to center, depreciating weight as we go to corners) improves the time taken by bot to win against the myopic player.

Depth = 5, Games = 10:

Time Without Move Ordering	Time With Move Ordering
27.48	16.48
29.38	14.84
33.24	14.88

Thus the time taken by the bot for each action is improved and overall game time is significantly reduced.