c)

[['B', 'B', 'B', 'B'], ['B', 'B', 'B', 'B'], ['B', 'B', 'B', 'W'], ['B', 'B', 'B', 'W']]

This answer means black will always win in the minimax algorithm scenario because black goes first. If white went first then the minimax algorithm would have white always win.

(14, ['B', 0, 2, 'W', 0, 1, 'B', 1, 0, 'W', 2, 3, 'B', 0, 0, 'W', -1, -1, 'B', 3, 1, 'W', 2, 0, 'B', 3, 0, 'W', -1, -1, 'B', 3, 2, 'W', -1, -1, 'B', 0, 3, 'W', -1, -1, 'B', 1, 3, ('W', -1, -1)])

Elapsed time: 3.93700003624

Terminal count: 38706

e)

4x4 alpha beta

(14, ['B', 0, 2, 'W', 0, 1, 'B', 1, 0, 'W', 2, 3, 'B', 0, 0, 'W', -1, -1, 'B', 3, 1, 'W', 2, 0, 'B', 3, 0, 'W', -1, -1, 'B', 3, 2, 'W', -1, -1, 'B', 0, 3, 'W', -1, -1, 'B', 1, 3, ('W', -1, -1)])

Elapsed time: 1.65299987793

Terminal count: 17320

trunc: 3780

4x4 without alpha-beta

(14, ['B', 0, 2, 'W', 0, 1, 'B', 1, 0, 'W', 2, 3, 'B', 0, 0, 'W', -1, -1, 'B', 3, 1, 'W', 2, 0, 'B', 3, 0, 'W', -1, -1, 'B', 3, 2, 'W', -1, -1, 'B', 0, 3, 'W', -1, -1, 'B', 1, 3, ('W', -1, -1)])

Elapsed time: 3.3180000782

Terminal count: 38706

5x5

[['W', 'B', 'B', 'B', 'B'], ['B', 'B', 'B', 'B', 'B'], ['B', 'B', 'B', 'B', 'B'], ['B', 'B', 'B', 'B', 'B'], ['B', 'B', ' ', 'W', 'W']]

(21, ['B', 1, 2, 'W', 2, 2, 'B', 3, 2, 'W', 1, 3, 'B', 0, 2, 'W', 2, 1, 'B', 2, 4, 'W', 0, 3, 'B', 2, 0, 'W', 1, 4, 'B', 0, 4, 'W', 3, 0, 'B', 2, 3, 'W', -1, -1, 'B', 4, 0, 'W', -1, -1, 'B', 4, 1, 'W', -1, -1, 'B', 3, 1, ('W', -1, -1)])

Elapsed time: 465.569000006

Terminal count: 4478485

trunc: 755778

This answer means black will always win in the minimax algorithm scenario because black goes first. If white went first then the minimax algorithm would have white always win. Black is not guaranteed to win in a fair game. However, our function finds the optimal values for black.