**Connect 4 project**

**Team member: Khaled mohamed abd el-kader**

**Minimax (Decision making algorithm)**

**Minimax** (sometimes **MinMax** or **MM**[[1]](https://en.wikipedia.org/wiki/Minimax" \l "cite_note-1)) is a decision rule used in [decision theory](https://en.wikipedia.org/wiki/Decision_theory" \o "Decision theory), [game theory](https://en.wikipedia.org/wiki/Game_theory" \o "Game theory), [statistics](https://en.wikipedia.org/wiki/Statistics" \o "Statistics) and [philosophy](https://en.wikipedia.org/wiki/Philosophy" \o "Philosophy) for *mini*mizing the possible [loss](https://en.wikipedia.org/wiki/Loss_function" \o "Loss function) for a worst case (*max*imum loss) scenario. Originally formulated for two-player [zero-sum](https://en.wikipedia.org/wiki/Zero-sum" \o "Zero-sum) [game theory](https://en.wikipedia.org/wiki/Game_theory" \o "Game theory), covering both the cases where players take alternate moves and those where they make simultaneous moves, it has also been extended to more complex games and to general decision-making in the presence of uncertainty.

### **Pseudocode**

The [pseudocode](https://en.wikipedia.org/wiki/Pseudocode" \o "Pseudocode) for the depth limited minimax algorithm is given below.

01 **function** minimax(node, depth, maximizingPlayer)

02 **if** depth = 0 **or** node is a terminal node

03 **return** the heuristic value of node

04 **if** maximizingPlayer

05 bestValue := −∞

06 **for each** child of node

07 v := minimax(child, depth − 1, FALSE)

08 bestValue := max(bestValue, v)

09 **return** bestValue

10 **else** *(\* minimizing player \*)*

11 bestValue := +∞

12 **for each** child of node

13 v := minimax(child, depth − 1, TRUE)

14 bestValue := min(bestValue, v)

15 **return** bestValue

*(\* Initial call for maximizing player \*)*

minimax(origin, depth, TRUE)