**EX.NO:4 DATE:4/9/2024**

**Reg.no:220701016**

**MINIMAX ALGORITHM**

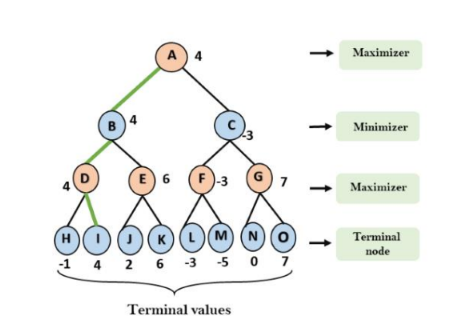
**AIM:**To implement MiniMax Algorith using python

• A simple example can be used to explain how the minimax algorithm works. We've included an example of a game-tree below, which represents a two-player game.

• There are two players in this scenario, one named Maximizer and the other named Minimizer.

• Maximizer will strive for the highest possible score, while Minimizer will strive for the lowest possible score.

• Because this algorithm uses DFS, we must go all the way through the leaves to reach the terminal nodes in this game-tree.

• The terminal values are given at the terminal node, so we'll compare them and retrace the tree till we reach the original state.  


**CODE**:

   import math

def minimax(depth, node\_index, is\_maximizer, scores, height):

if depth == height:

return scores[node\_index]

if is\_maximizer:

return max(minimax(depth + 1, node\_index \* 2, False, scores, height),

minimax(depth + 1, node\_index \* 2 + 1, False, scores, height))

else:

return min(minimax(depth + 1, node\_index \* 2, True, scores, height),

minimax(depth + 1, node\_index \* 2 + 1, True, scores, height))

def calculate\_tree\_height(num\_leaves):

return math.ceil(math.log2(num\_leaves))

scores = [3, 5, 6, 9, 1, 2, 0, -1]

tree\_height = calculate\_tree\_height(len(scores))

optimal\_score = minimax(0, 0, True, scores, tree\_height)

print(f"The optimal score is: {optimal\_score}")

OUTPUT:  
