# Minimax Algorithm in Python – Explanation

This document explains how the Minimax Algorithm works using the given Python code. Minimax is a decision-making algorithm used in Game Theory and Artificial Intelligence (AI) for games like Tic-Tac-Toe and Chess. It helps find the best possible move assuming both players play optimally.

## Python Code

import math  
  
def minimax(curDepth, nodeIndex, maxTurn, scores, targetDepth):  
 if curDepth == targetDepth:  
 return scores[nodeIndex]  
  
 if maxTurn:  
 return max(  
 minimax(curDepth + 1, nodeIndex \* 2, False, scores, targetDepth),  
 minimax(curDepth + 1, nodeIndex \* 2 + 1, False, scores, targetDepth)  
 )  
 else:  
 return min(  
 minimax(curDepth + 1, nodeIndex \* 2, True, scores, targetDepth),  
 minimax(curDepth + 1, nodeIndex \* 2 + 1, True, scores, targetDepth)  
 )  
  
scores = [3, 5, 2, 9, 3, 5, 2, 9]  
treeDepth = int(math.log(len(scores), 2))  
  
print("The optimal value is:", minimax(0, 0, True, scores, treeDepth))

## Explanation

1. The algorithm assumes two players: Maximizer and Minimizer.  
 - Maximizer tries to get the highest possible score.  
 - Minimizer tries to get the lowest possible score.  
  
2. The `scores` list represents all possible outcomes (leaf nodes) in the game tree.  
  
3. The `minimax` function takes several parameters:  
 - `curDepth`: the current depth in the tree.  
 - `nodeIndex`: the index of the current node.  
 - `maxTurn`: a boolean value; True for maximizer's turn, False for minimizer's turn.  
 - `scores`: list of possible scores at the leaf nodes.  
 - `targetDepth`: total depth of the game tree.  
  
4. The base case checks if the algorithm has reached the target depth (leaf node). If yes, it returns the corresponding score.  
  
5. If it's the maximizer's turn, the function calls itself recursively for both child nodes and returns the maximum of the two.  
  
6. If it's the minimizer's turn, it returns the minimum of the two child node values.  
  
7. The recursion continues until the base case is reached for all branches, and the best possible value is calculated at the root node.

## Output

The code prints the final optimal value after evaluating all possibilities:  
  
Output: The optimal value is: 9  
  
This means that if both players play optimally, the Maximizer can ensure a minimum guaranteed value of 9.