

week-3-assignment-2203a51510

February 19, 2024

```
[1]: import math

def minimax (curDepth, nodeIndex,
            maxTurn, scores,
            targetDepth):

    # base case : targetDepth reached
    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))

# Driver code
scores = [-1,4,2,6,-3,-5,0,7]

treeDepth = math.log(len(scores), 2)

print("The optimal value is : ", end = "")
print(minimax(0, 0, True, scores, treeDepth))
```

The optimal value is : 4

```
[2]: MAX, MIN = 1000, -1000

def minimax(depth, nodeIndex, maximizingPlayer,
            values, alpha, beta):
```

```

    if depth == 3:
        return values[nodeIndex]

    if maximizingPlayer:

        best = MIN

        for i in range(0, 2):

            val = minimax(depth + 1, nodeIndex * 2 + i,
                           False, values, alpha, beta)
            best = max(best, val)
            alpha = max(alpha, best)

            if beta <= alpha:
                break

        return best

    else:

        best = MAX

        for i in range(0, 2):

            val = minimax(depth + 1, nodeIndex * 2 + i,
                           True, values, alpha,
↪beta)

            best = min(best, val)
            beta = min(beta, best)

            if beta <= alpha:
                break

        return best
values = [-1,4,2,6,-3,-5,0,7]
print("The optimal value is :", minimax(0, 0, True, values, MIN, MAX))

```

The optimal value is : 4