

PROGRAM:

```
import math
def minimax (node, depth, is_maximizing):
    if depth == 0:
        return node.
```

```
    if is_maximizing:
```

```
        best_value = math.inf
```

```
        for child in get_children (node):
```

```
            value = minimax (child, depth-1, False)
```

```
            best_value = max (best_value, value)
```

```
        return best_value
```

```
    else:
```

```
        best_value = math.inf
```

```
        for child in get_children (node):
```

```
            value = minimax (child, depth-1, True)
```

```
            best_value = min (best_value, value)
```

```
        return best_value
```

```
def get_children (node):
```

```
    return node.get('children', [])
```

```
game_tree =
```

```
    {'value': 'A' ,
```

```
     'children': [
```

```
         {'value': 'B' , 'children': [
```

```
             {'value': 'D' , 'children': [], 'terminal_value': 3},
```

```
             {'value': 'E' , 'children': [], 'terminal_value': 6}
```

```
         ],
```

```
         {'value': 'C' , 'children': [
```

```
             {'value': 'F' , 'children': [], 'terminal_value': 1},
```

```
             {'value': 'G' , 'children': [], 'terminal_value': 4}
```

```
         ]
```

```
if __name__ == "__main__":
```

```
    best_score = minimax (game_tree, 2, True)
```

```
    print ("Best score for Maximizer (A) : {best_score}")
```

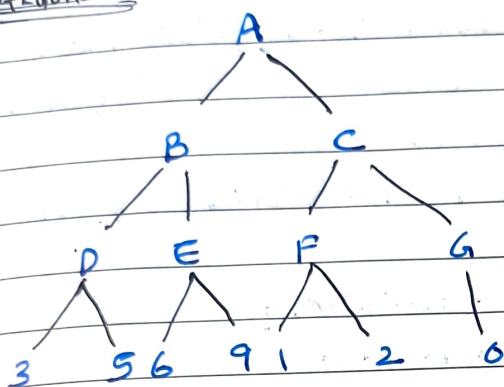


28/10/24

EXPNO 5

AIM: Implement MINIMAX algorithm in python.

ALGORITHM:



1. The function successively evaluates a tree
2. It takes node depth, depth of tree and a boolean if player is maximizing.
3. If it's a terminal node return node value.
4. The function gets child's nodes using get child nodes function.
5. Computes best score for maximizing A.

RESULT:

Thus the minimax algorithm has been implemented in python.