Ex.no:5

AIM:

To implement min max algorithm technique to determine the optimal move. The algorithm recursively evaluates game tree nodes until it identifies the move with best possible outcome for the current players.

PROGRAM:

```
def minimax(node, depth, maximizing_player):
    if depth == 0 or isinstance(node, int):
        return node.
    if maximizing_player:
        max_eval = float('-inf').
        for child in node:
            eval = minimax(child, depth-1, False)
            max_eval = max(max_eval, eval).
        return max_eval
    else:
        min_eval = float('inf')
        for child in node:
            eval = minimax(child, depth-1, True)
            min_eval = min(min_eval, eval)
        return min_eval.
```

```
game_tree = [[3, 5, 6],
             [9, 1, 2],
             [0, -1, 4]
            ]
initial_depth = 2
best_score = minimax(game_tree, initial_depth, True)
print("The optimal score for Maximizer is:", best_sco
```

output :-

The optimal score for maximizer is : 3.

result :-

The program is successfully executed and the output is verified.