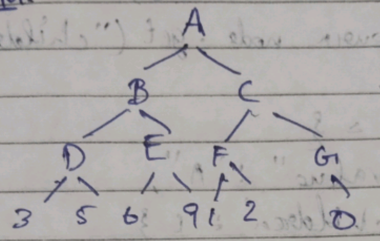
Exp-5

<u>Aim:</u>

Implement the maximum algorithm in python

<u>Algorithm:</u>



(i) The function recursively evaluate a tree
(ii) It takes node depth, depth of tree and boolean if player is maximum.
(iii) If its a terminal node return node value.
(iv) The function gets childs nodes asking the gets child node function.
(v) Computes best score for maximizing A.

<u>Program:</u>

```
def minmax (node ;depth , is_maximizing):
    if depth == 0
        return node
    if is_maximizing :
        best_value = - math.int
    for child in get_children (node):
        value = minimax (child, depth-1, fals
    best_value = max (best_value , value)
        return best_value
    else :
        best value = math.int
```

```
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":
                                                   [E]
        " ~~terminal value~~ , { } ; { 'value' : "E", "children'
                [ ] , "terminal_value : { } ] } ;

    { "value" : "C", "children" : [
        {"value" : "F", "children" : [ ], "terminal
                                                value" : [ ] }
        {"value" : "G", "children" [ ] , "terminal
                                       -value : ] } ) } }
                                                   } }
if __name__ == "__main__":
    best_score = minimax (game_tree, 4, True)
    print (f "Best score for maximize (A) :
```