Experiment-14: Alpha & Beta Pruning Algorithm

Aim:

To Print a Python Program to implement Alpha & Beta Pruning algorithm.

Program:

```
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

if __name__ == "__main__":

    values = [3, 5, 6, 9, 1, 2, 0, -1]
    print("The optimal value is :", minimax(0, 0, True, values, MIN,
MAX))

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

if __name__ == "__main__":

    values = [3, 5, 6, 9, 1, 2, 0, -1]
    print("The optimal value is :", minimax(0, 0, True, values, MIN,
MAX))
```

OUTPUT:

```
The optimal value is : 5
The optimal value is : 5
```

Result: The Code has been implemented Successfully.