console.aws.amazon.com/cloud9/ide/f57ef788c87f4dee8f3fa0ca75e5830e?#

alphabetapruning.py | minimaxalgorithm.py | tictactoeusingminimax

```python
def game():

    turn = 'X'
    count = 0


    for i in range(10):
        printBoard(theBoard)
        print("It's your turn," + turn + ".Move to which place?")

        move = input()

        if theBoard[move] == ' ':
            theBoard[move] = turn
            count += 1
        else:
            print("That place is already filled.\nMove to which place?")
            continue

        if count >= 5:
            if theBoard['7'] == theBoard['8'] == theBoard['9'] != ' ':
                printBoard(theBoard)
                print("\nGame Over.\n")
                print(" **** " +turn + " won. ****")
```

99:1 Python Spaces: 4

Command: tictactoeusingminimaxalgo.py

Runner: Python 3 CWD ENV

console.aws.amazon.com/cloud9/ide/f57ef788c87f4dee8f3fa0ca75e5830e?#

alphabetapruning.py | minimaxalgorithm.py | tictactoeusingminimax

```python
                break
            elif theBoard['2'] == theBoard['5'] == theBoard['8'] != ' ': # down the middle
                printBoard(theBoard)
                print("\nGame Over.\n")
                print(" **** " +turn + " won. ****")
                break
            elif theBoard['3'] == theBoard['6'] == theBoard['9'] != ' ': # down the right side
                printBoard(theBoard)
                print("\nGame Over.\n")
                print(" **** " +turn + " won. ****")
                break
            elif theBoard['7'] == theBoard['5'] == theBoard['3'] != ' ': # diagonal
                printBoard(theBoard)
                print("\nGame Over.\n")
                print(" **** " +turn + " won. ****")
                break
            elif theBoard['1'] == theBoard['5'] == theBoard['9'] != ' ': # diagonal
                printBoard(theBoard)
                print("\nGame Over.\n")
                print(" **** " +turn + " won. ****")
                break

        # If neither X nor O wins and the board is full, we'll declare the result as 'tie'.
        if count == 9:
```

99:1 Python Spaces: 4

Command: tictactoeusingminimaxalgo.py

Runner: Python 3 CWD ENV

console.aws.amazon.com/cloud9/ide/f57ef788c87f4dee8f3fa0ca75e5830e?#

alphabetapruning.py | minimaxalgorithm.py | tictactoeusingminimax

```python
                break

        # If neither X nor O wins and the board is full, we'll declare the result as 'tie'.
        if count == 9:
            print("\nGame Over.\n")
            print("It's a Tie!!")

        # Now we have to change the player after every move.
        if turn =='X':
            turn = 'O'
        else:
            turn = 'X'

    # Now we will ask if player wants to restart the game or not.
    restart = input("Do want to play Again?(y/n)")
    if restart == "y" or restart == "Y":
        for key in board_keys:
            theBoard[key] = " "

        game()

if __name__ == "__main__":
    game()
```

99:1 Python Spaces: 4

Command: tictactoeusingminimaxalgo.py

```
| |
-+-+-
| |
```

console.aws.amazon.com/cloud9/ide/f57ef788c87f4dee8f3fa0ca75e5830e?#

alphabetapruning.py | minimaxalgorithm.py | tictactoeusingminimax

```python
                break

        # If neither X nor O wins and the board is full, we'll declare the result as 'tie'.
        if count == 9:
            print("\nGame Over.\n")
            print("It's a Tie!!")
```

99:1 Python Spaces: 4

Command: tictactoeusingminimaxalgo.py

```
| |
-+-+-
| |
-+-+-
| |
It's your turn,X.Move to which place?
3
| |
-+-+-
| |
-+-+-
| |X
It's your turn,O.Move to which place?
1
| |
-+-+-
| |
-+-+-
O| |X
It's your turn,X.Move to which place?
```

console.aws.amazon.com/cloud9/ide/f57ef788c87f4dee8f3fa0ca75e5830e?#

alphabetapruning.py | minimaxalgorithm.py | tictactoeusingminimax

```
                break

        # If neither X nor O wins and the board is full, we'll declare the result as 'tie'.
        if count == 9:
            print("\nGame Over.\n")
            print("It's a Tie!!")

        # Now we have to change the player after every move.
```

99:1 Python Spaces: 4

alphabetapruning.py - Stc | minimaxalgorithm.py - Stc | tictactoeusingminimaxalg

Run Command: tictactoeusingminimaxalgo.py Runner: Python 3 CWD ENV

```
 | | 
-+-+-
 |o|x
-+-+-
o| |x
It's your turn,O.Move to which place?
9
x| |o
-+-+-
 |o|x
-+-+-
o| |x

Game Over.

 **** O won. ****
Do want to play Again?(y/n)n


Process exited with code: 0
Pane is dead
```

Type here to search 25°C ENG 09:20 07-03-2022

console.aws.amazon.com/cloud9/ide/f57ef788c87f4dee8f3fa0ca75e5830e?#

alphabetapruning.py | minimaxalgorithm.py | tictactoeusingminimax

```
MAX, MIN = 1000, -1000

# Returns optimal value for current player
#(Initially called for root and maximizer)
def minimax(depth, nodeIndex, maximizingPlayer,
            values, alpha, beta):

    # Terminating condition. i.e
    # leaf node is reached
    if depth == 3:
        return values[nodeIndex]

    if maximizingPlayer:

        best = MIN

        # Recur for left and right children
        for i in range(0, 2):
```

53:75 Python Spaces: 4

alphabetapruning.py - Stc | minimaxalgorithm.py - Stc | tictactoeusingminimaxalg

Run Command: tictactoeusingminimaxalgo.py Runner: Python 3 CWD ENV

```
Game Over.

 **** O won. ****
Do want to play Again?(y/n)n


Process exited with code: 0
Pane is dead
```

Type here to search 25°C ENG 09:21 07-03-2022

console.aws.amazon.com/cloud9/ide/f57ef788c87f4dee8f3fa0ca75e5830e?#

alphabetapruning.py | minimaxalgorithm.py | tictactoeusingminimax

```python
    # Terminating condition. i.e
    # leaf node is reached
    if depth == 3:
        return values[nodeIndex]

    if maximizingPlayer:

        best = MIN

        # Recur for left and right children
        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)

            # Alpha Beta Pruning
```

53:75 Python Spaces: 4

Command: tictactoeusingminimaxalgo.py

Runner: Python 3 CWD ENV

```
Game Over.

 **** O won. ****
Do want to play Again?(y/n)n


Process exited with code: 0
Pane is dead
```

console.aws.amazon.com/cloud9/ide/f57ef788c87f4dee8f3fa0ca75e5830e?#

alphabetapruning.py | minimaxalgorithm.py | tictactoeusingminimax

```python
    else:
        best = MAX

        # Recur for left and
        # right children
        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)

            # Alpha Beta Pruning
            if beta <= alpha:
                break

        return best

# Driver Code
```

53:75 Python Spaces: 4

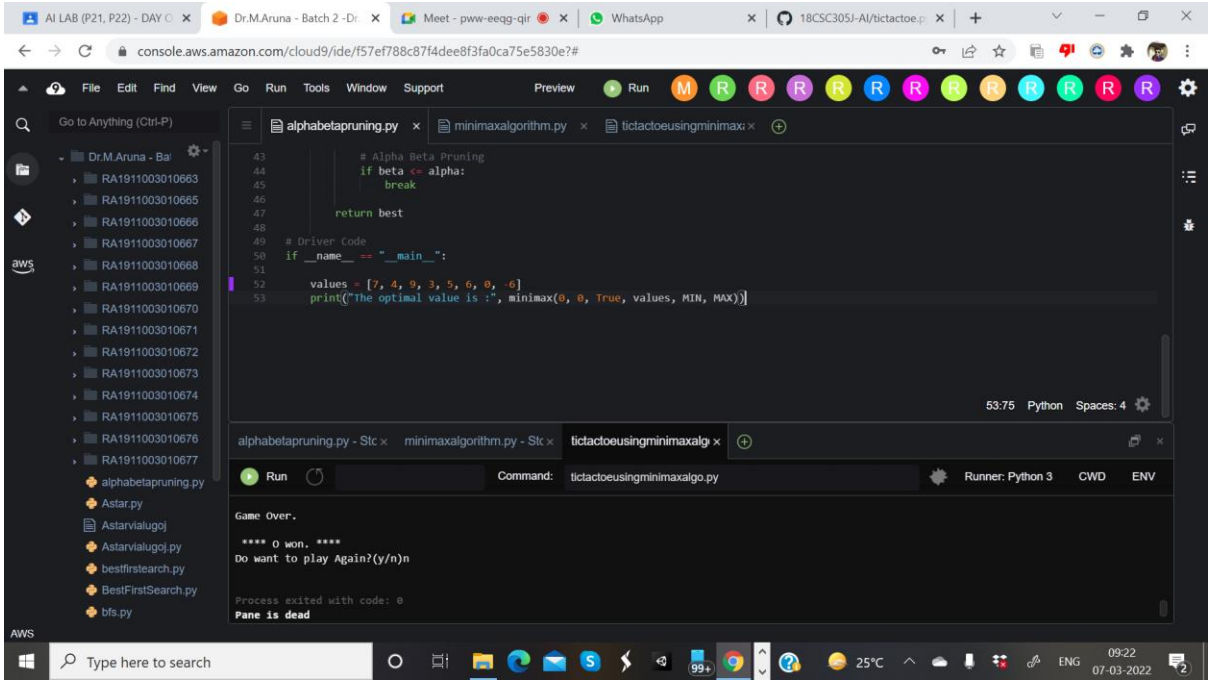
Command: tictactoeusingminimaxalgo.py

Runner: Python 3 CWD ENV

```
Game Over.

 **** O won. ****
Do want to play Again?(y/n)n


Process exited with code: 0
Pane is dead
```


console.aws.amazon.com/cloud9/ide/f57ef788c87f4dee8f3fa0ca75e5830e?#
alphabetapruning.py
minimaxalgorithm.py
tictactoeusingminimax
# Alpha Beta Pruning
if beta <= alpha:
break
return best
# Driver Code
if __name__ == "__main__":
values = [7, 4, 9, 3, 5, 6, 0, -6]
print("The optimal value is :", minimax(0, 0, True, values, MIN, MAX))
53:75 Python Spaces: 4
Command: tictactoeusingminimaxalgo.py
Runner: Python 3
Game Over.
**** O won. ****
Do want to play Again?(y/n)n
Process exited with code: 0
Pane is dead