A.HariChandana
RA1911026010009
CSE-AI&ML

AIM: To solve the 4 queens problem using the Backtracking technique.

ALGORITHM: Backtracking algorithm

1) Start in the leftmost column.
2) If all queens are placed
   return true
3) Try all rows in the current column.
 Perform the following operations for every tried row.
   a) If the queen can be placed safely in this row then mark this [row, column] as part of the solution and recursively check if placing queen here leads to a solution.
   b) If placing the queen in [row, column] leads to a solution then return true.
   c) If placing queen doesn't lead to a solution then unmark this [row, column] (Backtrack) and go to step (a) to try other rows.
3) If all rows have been tried and nothing worked,
  return false to trigger backtracking.

PROGRAM :

```
global N
N = 4

def printSolution(board):
        for i in range(N):
                for j in range(N):
                        print (board[i][j], end = " ")
                print()
```

A.HariChandana
RA1911026010009

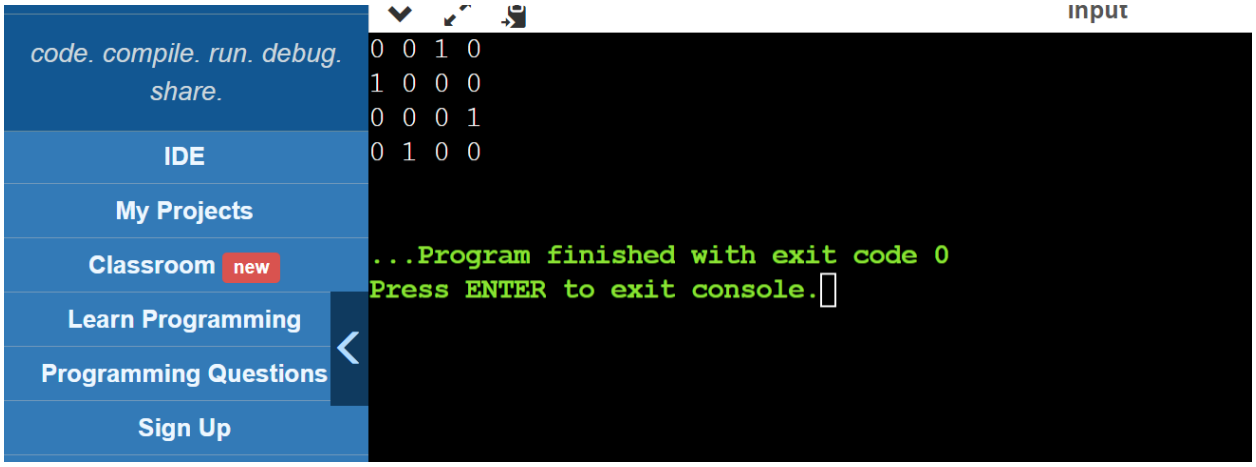
```
def isSafe(board, row, col):

    for i in range(col):
        if board[row][i] == 1:
            return False
    for i, j in zip(range(row, -1, -1),
                    range(col, -1, -1)):
        if board[i][j] == 1:
            return False
    for i, j in zip(range(row, N, 1),
                    range(col, -1, -1)):
        if board[i][j] == 1:
            return False

    return True

def solveNQUtil(board, col):

    if col >= N:
        return True

    for i in range(N):

        if isSafe(board, i, col):

            board[i][col] = 1

            if solveNQUtil(board, col + 1) == True:
                return True
```

A.HariChandana
RA1911026010009

```
            board[i][col] = 0

        return False

def solveNQ():
        board = [ [0, 0, 0, 0],
                  [0, 0, 0, 0],
                  [0, 0, 0, 0],
                  [0, 0, 0, 0] ]

        if solveNQUtil(board, 0) == False:
            print ("Solution does not exist")
            return False

        printSolution(board)
        return True
solveNQ()
```

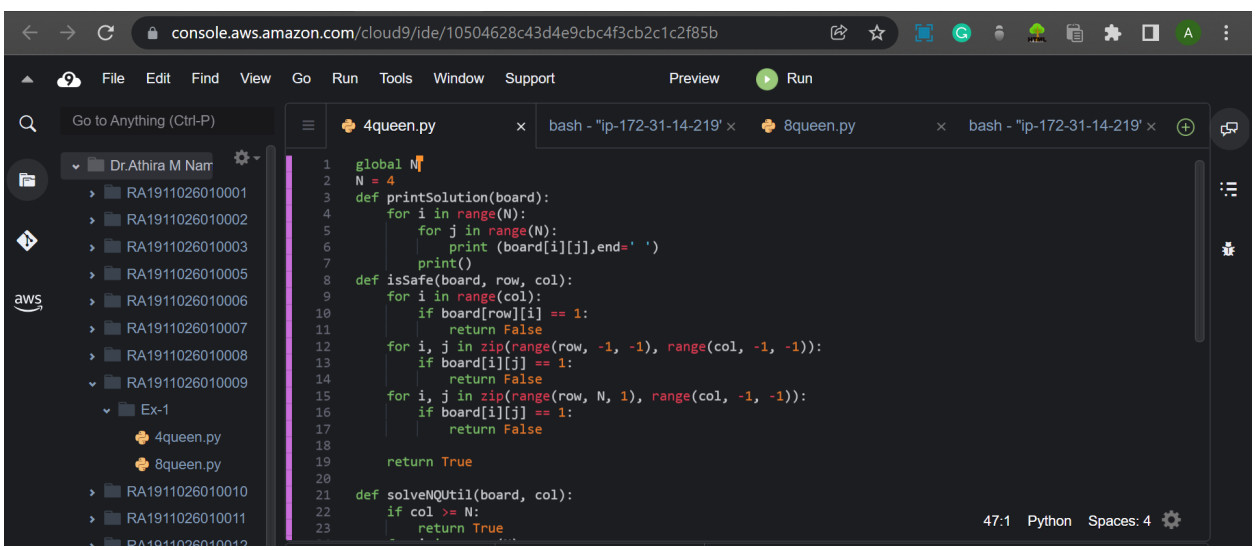
A.HariChandana
RA1911026010009

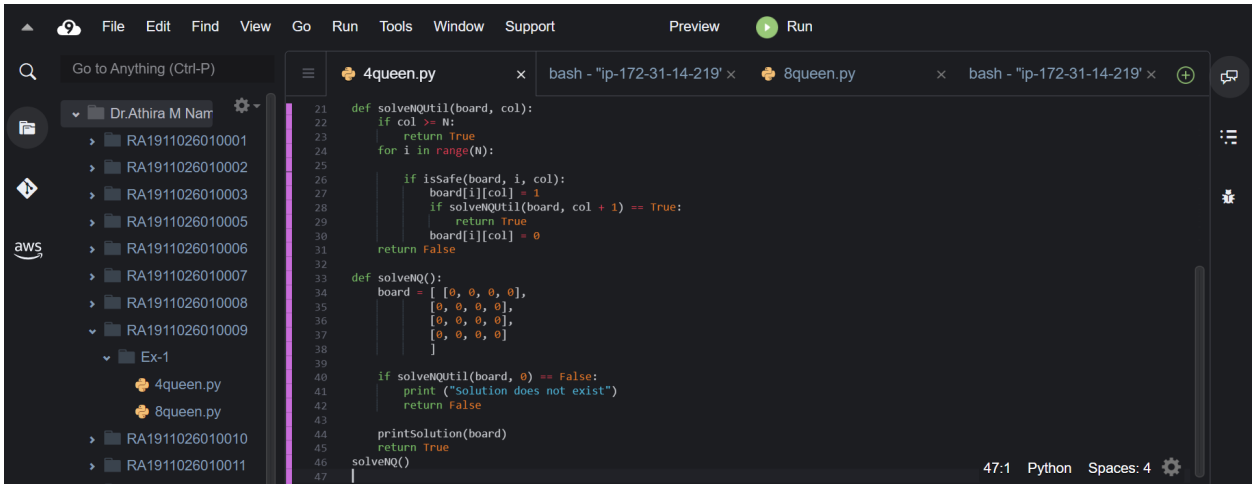
SCREENSHOT OF THE OUTPUT:



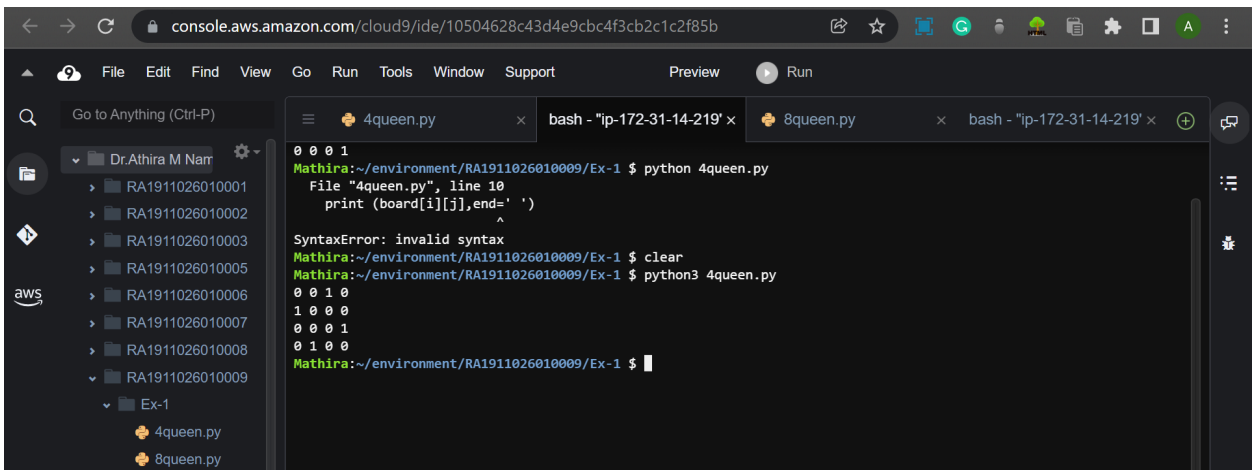
OUTPUTS IN AWS :

4-QUEENS SOURCE CODE SCREENSHOT:

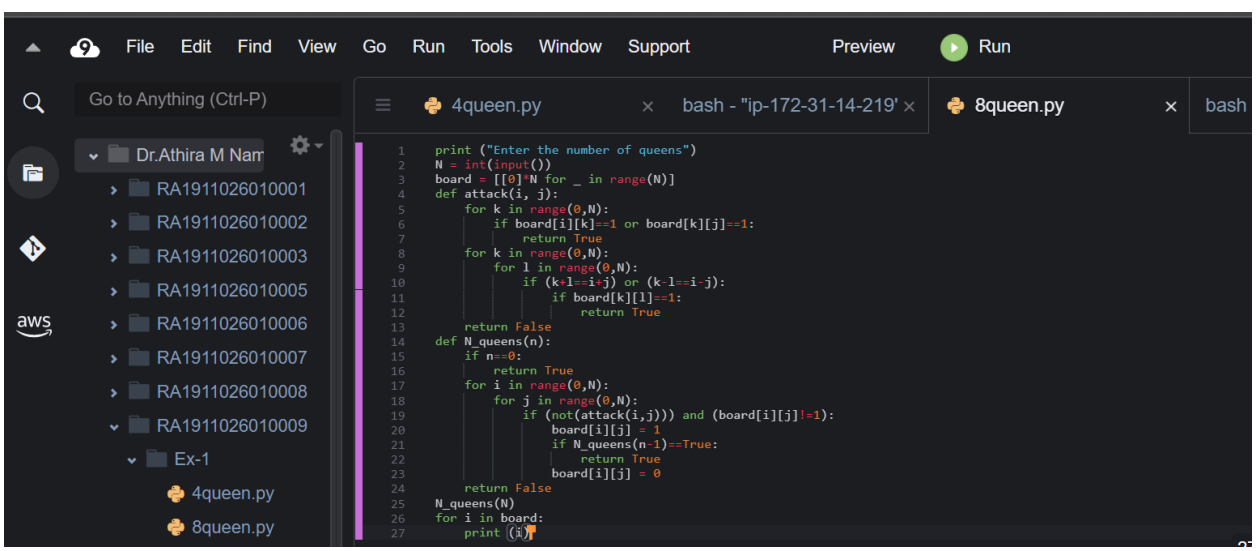




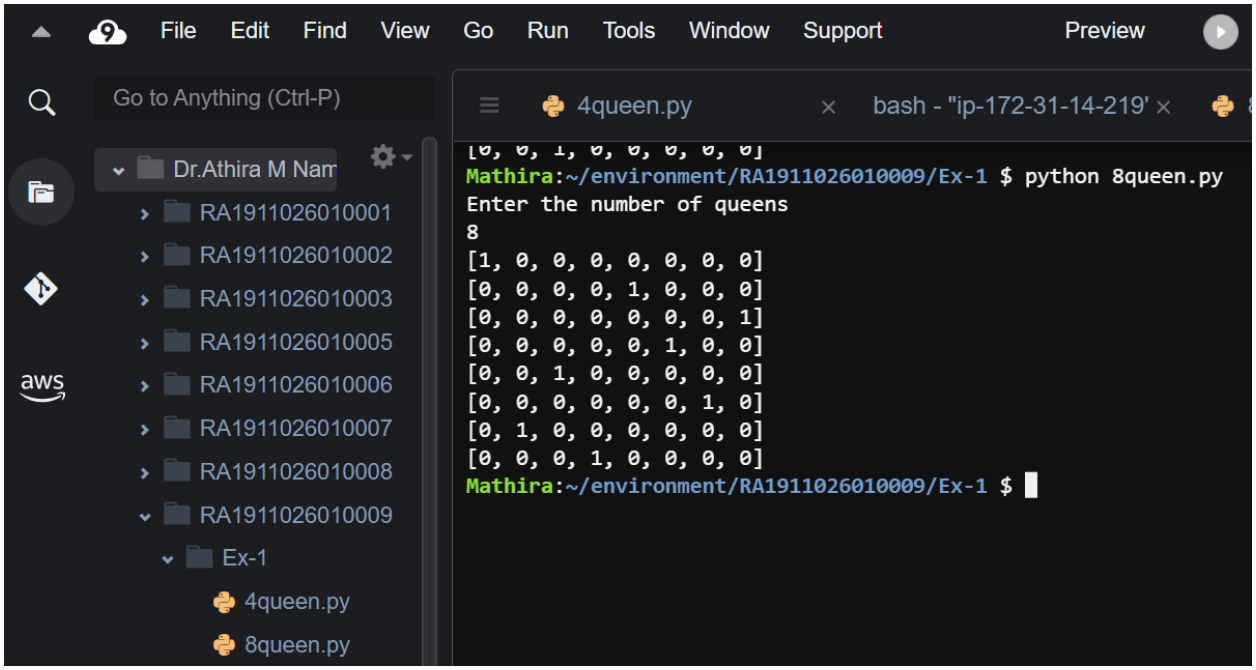
OUTPUT OF 4-QUEEN:



8-QUEEN SOURCE CODE SCREENSHOT:



OUTPUT OF 8-QUEEN PROBLEM:



RESULT: Thus, the implementation of 4 queens and 8-queens has been done using the Backtracking algorithm.