DAA Report Week 12

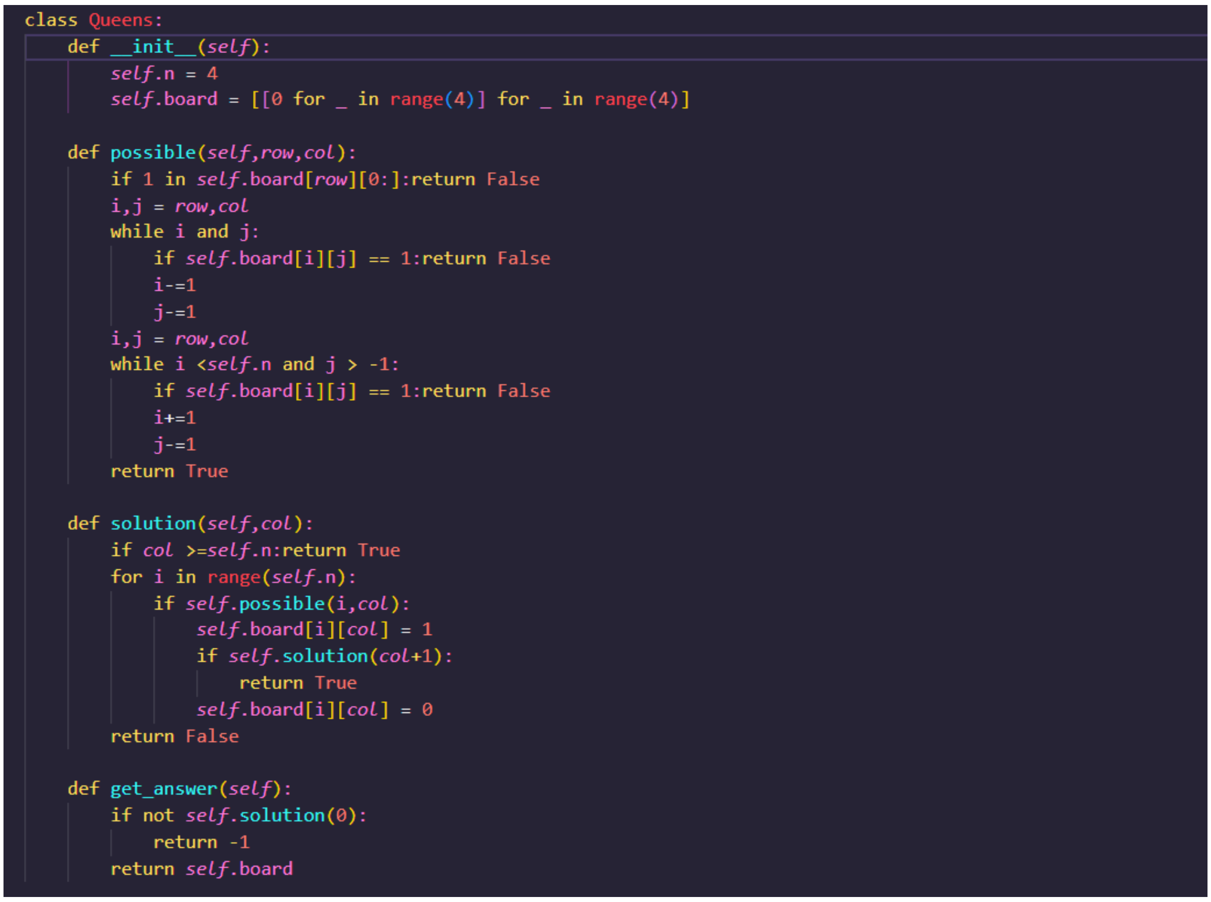
N Queens Problem (4X4 Grid)

Shashank K

SE21UCSE198

CSE 3

Program



```python
class Queens:
    def __init__(self):
        self.n = 4
        self.board = [[0 for _ in range(4)] for _ in range(4)]

    def possible(self,row,col):
        if 1 in self.board[row][0:]:return False
        i,j = row,col
        while i and j:
            if self.board[i][j] == 1:return False
            i-=1
            j-=1
        i,j = row,col
        while i <self.n and j > -1:
            if self.board[i][j] == 1:return False
            i+=1
            j-=1
        return True

    def solution(self,col):
        if col >=self.n:return True
        for i in range(self.n):
            if self.possible(i,col):
                self.board[i][col] = 1
                if self.solution(col+1):
                    return True
                self.board[i][col] = 0
        return False

    def get_answer(self):
        if not self.solution(0):
            return -1
        return self.board
```

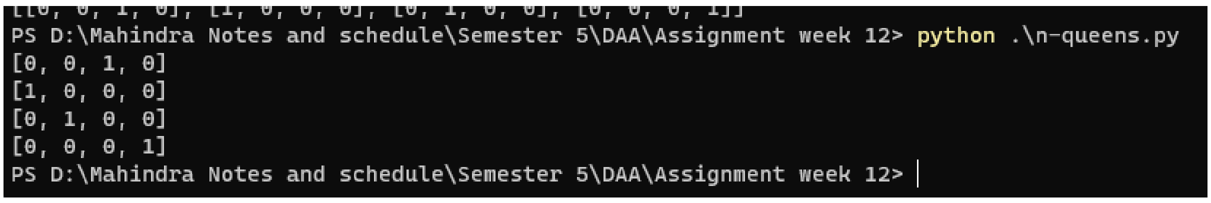
Methods:

possible : Checks whether a queen can be placed on grid at point (row,col). Checking makes sure that there are no attacking queens present.

Solution : An iterative method that implements the n-queens algorithm. We try placing in each row one by one in the respective column.

Get_answer : Driver method that checks if it solution is possible and returns respective board configuration.

Output:



```
[[0, 0, 1, 0], [1, 0, 0, 0], [0, 1, 0, 0], [0, 0, 0, 1]]
PS D:\Mahindra Notes and schedule\Semester 5\DAA\Assignment week 12> python .\n-queens.py
[0, 0, 1, 0]
[1, 0, 0, 0]
[0, 1, 0, 0]
[0, 0, 0, 1]
PS D:\Mahindra Notes and schedule\Semester 5\DAA\Assignment week 12> |
```

Time complexity : O(N!)   Space Complexity : $O(N^2)$

where N is the board size

THE END