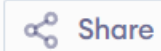


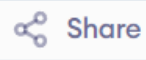
main.py



Run

```
1 N = 8
2
3 def printSolution(board):
4     for row in board:
5         for i in range(N):
6             print("Q" if row[i] else ".", end=" ")
7         print()
8
9 def isSafe(board, row, col):
10     for i in range(row):
11         if board[i][col]:
12             return False
13     for i, j in zip(range(row - 1, -1, -1), range(col - 1, -1, -1)):
14         if board[i][j]:
15             return False
16     for i, j in zip(range(row - 1, -1, -1), range(col + 1, N)):
17         if board[i][j]:
18             return False
19     return True
20
21 def solve(board, row):
22     if row == N:
23         printSolution(board)
24         return True
25     for col in range(N):
26         if isSafe(board, row, col):
```

main.py



Run

```
13- for i, j in zip(range(row - 1, -1, -1), range(col - 1, -1, -1)):
14-     if board[i][j]:
15-         return False
16- for i, j in zip(range(row - 1, -1, -1), range(col + 1, N)):
17-     if board[i][j]:
18-         return False
19- return True
20
21- def solve(board, row):
22-     if row == N:
23-         printSolution(board)
24-         return True
25-     for col in range(N):
26-         if isSafe(board, row, col):
27-             board[row][col] = 1
28-             if solve(board, row + 1):
29-                 return True
30-             board[row][col] = 0
31-     return False
32
33- def eightQueens():
34-     board = [[0 for _ in range(N)] for _ in range(N)]
35-     solve(board, 0)
36
37- eightQueens()
38
```

## Output

```
Q . . . . .
. . . . Q . .
. . . . . Q
. . . . . Q .
. . Q . . . .
. . . . . Q .
. Q . . . . .
. . . Q . . .
```

=== Code Execution Successful ===