# 8-Queens Problem Solver Using Backtracking (Python)

## Python Code:

N = 8  
  
def print\_solution(board):  
 for row in board:  
 print(" ".join("Q" if col else "." for col in row))  
 print()  
  
def is\_safe(board, row, col):  
 for i in range(row):  
 if board[i][col]:  
 return False  
 i, j = row, col  
 while i >= 0 and j >= 0:  
 if board[i][j]:  
 return False  
 i -= 1  
 j -= 1  
 i, j = row, col  
 while i >= 0 and j < N:  
 if board[i][j]:  
 return False  
 i -= 1  
 j += 1  
 return True  
  
def solve\_nqueens(board, row):  
 if row == N:  
 print\_solution(board)  
 return True  
  
 for col in range(N):  
 if is\_safe(board, row, col):  
 board[row][col] = 1  
 if solve\_nqueens(board, row + 1):  
 return True  
 board[row][col] = 0  
  
 return False  
  
def main():  
 board = [[0 for \_ in range(N)] for \_ in range(N)]  
 if not solve\_nqueens(board, 0):  
 print("No solution exists.")  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 main()

## Sample Output:

. Q . . . . . .  
. . . . Q . . .  
. . . . . . Q .  
. . Q . . . . .  
. . . . . Q . .  
Q . . . . . . .  
. . . Q . . . .  
. . . . . . . Q