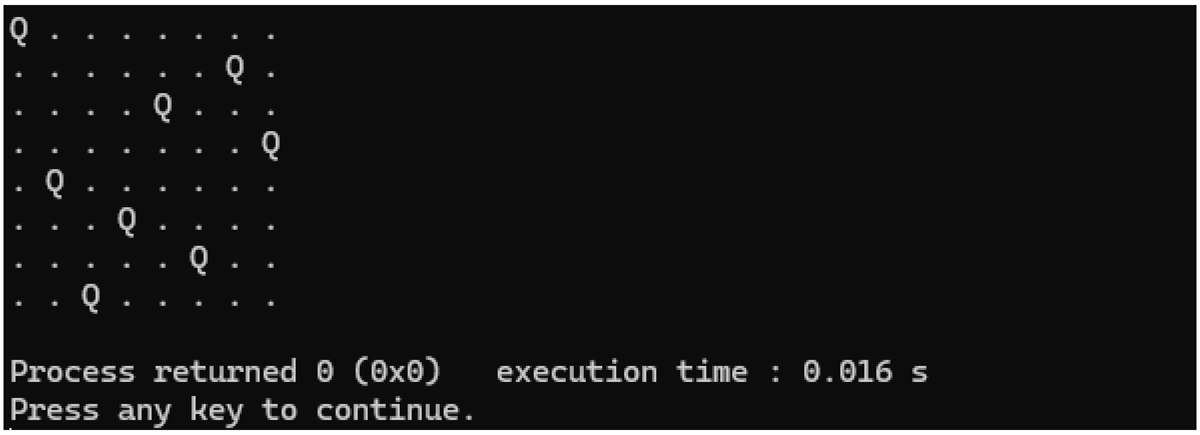
N-Queens Problem

```
#include <stdio.h>
#include <stdbool.h>

#define N 8

void printSolution(int board[N][N]) {
    for (int i = 0; i < N; i++) {
        for (int j = 0; j < N; j++)
            printf("%s ", board[i][j] ? "Q" : ".");
        printf("\n");
    }
}

bool isSafe(int board[N][N], int row, int col) {
    for (int i = 0; i < col; i++)
        if (board[row][i])
            return false;

    for (int i = row, j = col; i >= 0 && j >= 0; i--, j--)
        if (board[i][j])
            return false;

    for (int i = row, j = col; i < N && j >= 0; i++, j--)
        if (board[i][j])
            return false;

    return true;
}
```

```
bool solveNQUtil(int board[N][N], int col) {
    if (col >= N)
        return true;

    for (int i = 0; i < N; i++) {
        if (isSafe(board, i, col)) {
            board[i][col] = 1;

            if (solveNQUtil(board, col + 1))
                return true;

            board[i][col] = 0;
        }
    }
    return false;
}

bool solveNQ() {
    int board[N][N] = {0};

    if (!solveNQUtil(board, 0)) {
        printf("Solution does not exist\n");
        return false;
    }

    printSolution(board);
    return true;
}
```

```
int main() {
    solveNQ();
    return 0;
}
```

OUTPUT:



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

Process returned 0 (0x0)   execution time : 0.016 s
Press any key to continue.
```