27. The n-queens puzzle is the problem of placing a queens on an nxn chessboard such that no two queens attack each other. Given an integer n, retum all distinct solutions to the n-queens puzzle. You may retum the answer in any order. Write a program for the same.

CODE

#include <stdio.h>

#define N 10

int board[N][N];

void initializeBoard(int n)

{

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; j++)

{

board[i][j] = 0;

}

}

}

void printBoard(int n)

{

for (int i = 0; i < n; i++)

{

for (int j = 0; j < n; j++)

{

printf("%c ", board[i][j] ? 'Q' : '.');

}

printf("\n");

}

printf("\n");

}

int isSafe(int row, int col, int n)

{

for (int i = 0; i < col; i++)

{

if (board[row][i])

{

return 0;

}

}

for (int i = row, j = col; i >= 0 && j >= 0; i--, j--)

{

if (board[i][j])

{

return 0;

}

}

for (int i = row, j = col; i < n && j >= 0; i++, j--)

{

if (board[i][j])

{

return 0;

}

}

return 1;

}

void solveNQueens(int col, int n)

{

if (col >= n)

{

printBoard(n);

return;

}

for (int i = 0; i < n; i++)

{

if (isSafe(i, col, n))

{

board[i][col] = 1;

solveNQueens(col + 1, n);

board[i][col] = 0;

}

}

}

int main()

{

int n;

printf("Enter the value of n: ");

scanf("%d", &n);

if (n <= 0 || n > N)

{

printf("Invalid board size.\n");

return 1;

}

initializeBoard(n);

solveNQueens(0, n);

return 0;

}

OUTPUT

