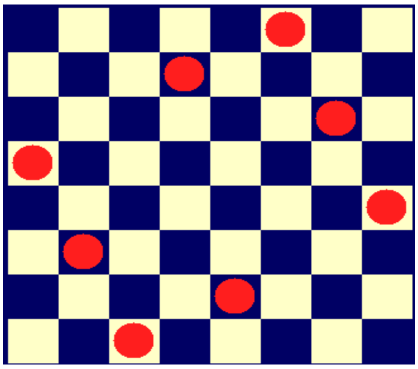
N-Queens Problem

The eight queens puzzle is the problem of placing eight chess queens on 8×8 chessboard so that no two queens attack each other. Thus, a solution requires that no two queens share the same row, column, or diagonal. The eight queens puzzle is an example of the more general n-queens problem of placing n queens on an n×n chessboard, where solutions exist only for n = 1 or n ≥ 4.
The problem can be quite computationally expensive as there are 4,426,165,368 (i.e., 64 choose 8) possible arrangements of eight queens on a 8×8 board, but only 92 solutions.
In chess, a queen can move as far as she pleases, horizontally, vertically, or diagonally. A chess board has 8 rows and 8 columns. The standard 8 by 8 Queen's problem asks how to place 8 queens on an ordinary chess board so that none of them can hit any other in one move.
It turns out that there are 12 essentially distinct solutions. (Two solutions are not essentially distinct if you can obtain one from another by rotating your chess board, or placing it in front of a mirror, or combining these two operations.)



The program finds solutions by starting with a queen in the top left corner of the chess board. It then places a queen in the second column and moves it until it finds a place where it cannot be hit by the queen in the first column. It then places a queen in the third column and moves it until it cannot be hit by either of the first two queens. Then it continues this process with the remaining columns. If there is no place for a queen in the current column the program goes back to the preceding column and moves the queen in that column. If the queen there is at the end of the column it removes that queen as well and goes to the preceding column. If the current column is the last column and a safe place has been found for the last queen, then a solution of the puzzle has been found. If the current column is the first column and its queen is being moved off the board then all possible configurations have been examined, all solutions have been found, and the algorithm terminates.

Q1	x	x	
	x	x	
	Q2	x	
		x	

1

Q1	x	x	x
	x	Q3	x
	x		x
	Q2		x

2

Q1	x	x	
	x	x	
	x	x	
	Q2	x	

3

x	x	Q3	x
Q1	x		x
	x		Q4
	Q2		

4

Solution 1

x	x	Q3	x
Q1	x		x
	x		x
	Q2		x

5

Q1	x	x	
	x	Q3	
	x		
	Q2		

6

Q1	x	x	x
	x	Q3	x
	x		x
	Q2		x

7

Q1	x	x	
	x	x	
	x	x	
	Q2	x	

8

x	Q2	x	x
x		x	Q4
Q1		x	
		Q3	

9

Solution 2

x	Q2	x	x
x		x	x
Q1		x	x
		Q3	x

10

x	x		
x	x		
Q1	x		
	x		

11

x	Q2	x	x
x		x	x
x		Q3	x
Q1			x

12

x	Q2	x	
x		x	
x		x	
Q1		x	

13

Program C:

```
#define MAX 100
```

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

int n; //number of queens
int x[MAX] ; //array which stores columns of queens

void read()
{
    printf("Enter number of queens: ");
    scanf("%d",  &n);
}

int place(int k , int i)
{
    for (int j = 1; j <= k-1; j++)
    {
      if (x[j] == i || abs(x[j]-i) == (k-j))
       {
          return 0;
       }
    }
    printf("Can be placed\n");
    return 1;
}

void nqueen(int k)
{
    for (int i = 1 ; i <= n ; i++)
    {
        if (place(k, i))
          {
            x[k] = i;
            if (k==n)
               {
                   //print solution
                   for (int j = 1 ; j <= n ; j++)
                       printf("%d    ", x[j]);
                   printf(" \n");
               }
            else
               nqueen(k+1);
          }
    } //end for
}//end nqueen

void main()
{
       read();
       nqueen(1);
}
```