The n queens problem

By the “n queens problem” we mean the problem of placing n queens on an nXn “chessboard” in such a way that no queen can capture any other on the next move. In class we solved the “8 queens” problem.

**Write a function** that inputs an integer n and returns the **number of solutions** to the “n queens” problem. Your function should use the one dimensional representation for the board, the algorithm we discussed in class, and no gotos.

**Test your function** with a main program that prompts the user for an integer n. The main program then calls the function n times, once for each number from 1 – n, and then prints the number of solutions to each of these problems, one on a line.

For example, if you enter n=5 your program should output:

1. There are _____ solutions to the 1 queens problem.
2. There are _____ solutions to the 2 queens problem.
3. There are _____ solutions to the 3 queens problem.
4. There are _____ solutions to the 4 queens problem.
5. There are _____ solutions to the 5 queens problem.

Now, since each time through the loop you will need an array q of a different length, you will need to allocate the array off of the heap (which we mentioned in class) and not the run-time stack. To do this you use the “new operator” to request the heap to dynamically allocate the memory for you.

For example, to get a one dimensional array of integers of size n and called q, we use the following syntax:

```cpp
int* q = new int[n];
```

This allocates the array for us **dynamically**, at run-time. After this we can use the array q just as if it had been declared “normally” and it has n elements denoted q[0] through q[n-1].

When we no longer need the memory that was allocated to the array, we write:

```cpp
delete [ ] q;
```

For this problem, you pass the required length for the array q to your function, which then allocates q dynamically (using “new”) and uses it to hold the solutions for the given size. Each time the function exits, you must deallocate q by calling delete [ ] q.

**For this problem, test your program for n=8.**

For some more details on how to use new and delete, go to the video section of the course web site and see the video called “Dynamically allocated arrays.”