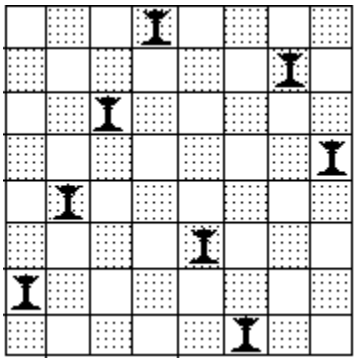
CSCI561 – Introduction to Artificial Intelligence
Instructor: Dr. K. Narayanaswamy

Assignment 2 - n Queen
Due: 10/03/2011 11:59:59pm
Electronic Submission (on Blackboard)

1. Problem Definition

The n queen puzzle is the problem of putting n chess queens on an n*n chessboard such that none of them is able to capture any other using the standard chess queen's moves. The color of the queens is meaningless in this puzzle, and any queen is assumed to be able to attack any other. Thus, a solution requires that no two queens share the same row, column, or diagonal.
The figure below gives an example of 8 queen puzzle on 8 * 8 chessboard.



Input:-

There will be two parameters of your program. The first one is the size of the chessboard (0<n<=8), and the second one is the name of the output file. Thus, the command line will be:
puzzle n outputFile

Output:-

Results of your program should be written into files. The first line in the output file shows the number of queens. Then all the solutions will be listed one in a line. Each solution is represented by the coordinates of the queens. Also, please give the total number of solutions at the end of output file. Here is an example of the output file:

Number of Queens: 8
Solution 1: (1, 2), (2, 4), (3, 6), (4, 8), (5, 3), (6, 1), (7, 7), (8, 5)
Solution 2: (1, 3), (2, 1), (3, 7), (4, 5), (5, 8), (6, 2), (7, 4), (8, 6)
…
…
…
Solution 92: …
Total number of solutions: 92

---

2. Algorithm Requirement

A Constraint Satisfaction Problem consists of 3 components

1) A set of variables.
2) A set of values for each of the variables.
3) A set of constraints between various collections of variables.

You must represent the n-queen puzzle as a constraint satisfaction problem.

You might first try to find a solution by hand: - http://www.hbmeyer.de/backtrack/achtdamen/eight.htm#up .

3. Coding Requirement

Your program must be written in C++ or Java.

You may use the Standard Libraries (e.g. STL for C++ or Java's standard container libraries), you may not use any 3rd party objects/algorithms without prior permission from the course TA or Professor (don't go download code that runs search algorithms).

No collaboration will be allowed on this project. These must reflect just your own work, with no outside help (except for questions asked of the instructor and TA). This explicitly includes no use of any code from elsewhere (including the Internet) without explicit permission from the course Instructor or TA.

Your program must compile and run on Aludra (see: http://www.usc.edu/its/web/getting started/ppages.html if you are unfamiliar with Aludra).

4. Submission Guidelines

a) General Coding Guidelines

For the C++ programmers:

Make sure that your source code can be compiled and executed on aludra.usc.edu. Please provide a README file with instructions to run your programs and a Makefile to compile where possible.

For the JAVA programmers:

All JAVA classes should be in package CS561A2.<LastName> and the source files should be in src/CS561A2/<LastName>. Please provide a README file with instructions to run your programs.

For Submission:

Place all the required files in a single directoryZip all the files in your submission into a single file called: - Firstname_LastName_A1.zip

Only Zip format will be accepted.

Please do not use any other processes or tools.

b) **Execution Details**

puzzle n outputFile
or

java –cp CLASSPATH CS561A2.<LastName>.puzzle n outputFile

where 'outputFile' is the text file which will enclose the solution as described above.

c) **README.txt**

This file must contain the following information
– Your name (First and Last)
– Your USC Student number
– Your USC email address
– Implementation Details(Steps of Algorithm)
– Any additional information you want us to know when grading your work.

5. **Grading Guidelines**

• Incorrect File names and extensions (case sensitive): - [- 5 points]

• Your program will be tested with 4 different board sizes. Grades will be assigned based on the following:

Correct listing of **all the Solutions** for each input board size: - 25 * 4 [100 points]

• Missing README file: - [-10 points]
(even if lots of comments in code)

• Your code must be robust. If your program crashes or fails to terminate in reasonable time on any test case, then it fails that test. If your program terminates gracefully, more partial credit might be awarded.

Please make sure ALL source files are submitted in the zip file you submit. Errors in submission will be assessed –25 points.

A program that does not compile as submitted will be given 0 points. None of your other submissions will be graded if the program does not compile.

Only your FINAL submission will be graded.

For policies on late submissions, please see the Syllabus from the course home page. These policies will be enforced with no exceptions.