UNIVERSITY OF LONDON
IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

EXAMINATIONS 2002

BEng Honours Degree in Computing Part I
MEng Honours Degrees in Computing Part I
for Internal Students of the Imperial College of Science, Technology and Medicine

This paper is also taken for the relevant examinations for the Associateship of the City and Guilds of London Institute

PAPER C121

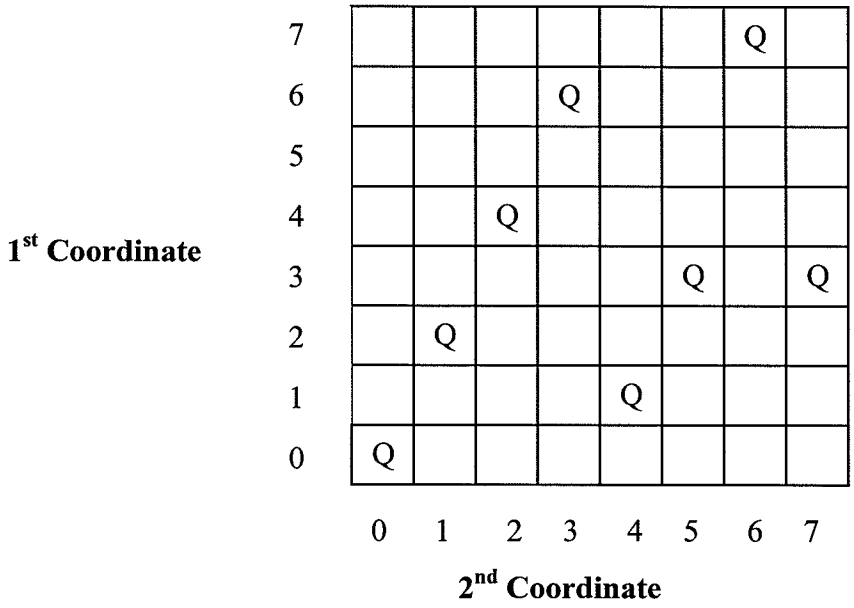
PROGRAMMING I

Monday 25 March 2002, 11:30
Duration: 90 minutes

Answer ONE question

Paper contains 1 question
Calculators not required

Placing eight queens on a chessboard such that no two queens can capture each other is called the **Eight Queens Problem.** A queen can capture another queen if they are both in the same row, in the same column or on a diagonal. The picture below shows a chessboard with eight queens placed on it. It isn't a solution, though, because the queens at indices (2,1) and (7,6) can capture each other since they are on the same diagonal. Also, the queens at indices (3,5) and (3,7) can capture each other because they are on the same row.



The queens will be held in a one-dimensional array of integers such that the i^{th} element of the queens array will contain –1 if there is no queen in column i and the row number of the queen if one exists in column i. (Note that there is no way to represent a board that has two queens in the same column.) A chessboard with no queens on it would be declared as:

```
int [ ] queens = {-1,-1,-1,-1,-1,-1,-1,-1};
```

The chessboard above would be declared as:

```
int [ ] queens = {0,2,4,6,1,3,7,3};
```

A chessboard with a queen at the bottom left corner and another at the top right corner would be:

```
int [ ] queens = {0,-1,-1,-1,-1,-1,-1,7};
```

An example of a chessboard that solves the eight queens problem is the following:

```
int [ ] queens = {2,4,1,7,0,6,3,5};
```

1 Answer using Kenya or Java with pre and post conditions (in logic or English) where appropriate. There will be a 5 mark penalty for non-compilation.

a Write a method `getQueens` which prompts for each queen in turn, in the following manner and returns an array of integers containing all eight queens. You can assume that the user will only type in numbers between 0 and 7 inclusive.

```
Please type in the row number for each column.
Column 0 ---> 0
Column 1 ---> 2
etc.
```

b For this question you are to assume that there are eight queens on the board. Each method required takes as an argument an array of integers representing eight queens, as described above, and returns a `boolean`.

i) Write a predicate `hasBadRow` which returns `true` if and only if there are two or more queens in the same row.

ii) Write a predicate `hasBadDiagonal` which returns `true` if and only if there are two or more queens on the same diagonal.

Hint: *look at the coordinates of all squares on a given diagonal.* You may find it easier if you first deal with all the possible diagonals in one direction and then all the possible diagonals in the other direction.

iii) Write a predicate `isSolution` that returns `true` if and only if there are eight queens such that no two can capture each other.

c Write a method `printBoard` which takes as an argument an array of integers representing the placed queens, as described, and prints the board on the screen. There should be two spaces between each square. If the square contains a Queen then `Q` should be printed. If a square is empty then a full stop (e.g. '`.`') should be printed. So for the example, `printBoard` should output on the screen:

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

The three parts carry, respectively, 10%, 70%, 20% of the marks.