Institut für Systems Engineering
Fachgebiet Simulation und Modellierung
Programmieren II

Leibniz Universität Hannover

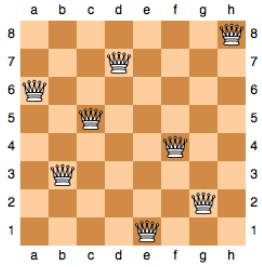
Vorlesung:
Priv.-Doz. Dr.-Ing. Matthias Becker
eMail: xmb@sim.uni-hannover.de

Übungsbetrieb:
B.Sc. Christian Kater
eMail: kater@sim.uni-hannover.de

Aufgabenblatt vom 15.04.2014
Abgabe bis **22.04.2014, 03:00**

1. Hausübung

Algorithmen, Abstrakte Klassen



Eine Lösung des 8-Damen-Problems

In dieser Aufgabe sollen verschiedene Typen von Algorithmen zur Lösung des Damenproblems implementiert werden.

Das Damenproblem ist eine schachmathematische Aufgabe. Es sollen jeweils acht Damen auf einem Schachbrett so aufgestellt werden, dass keine zwei Damen einander nach den Schachregeln schlagen können. Die Figurenfarbe wird dabei ignoriert, und es wird angenommen, dass jede Figur jede andere angreifen könnte. Oder anders ausgedrückt: Es sollen sich keine zwei Damen die gleiche Reihe, Linie oder Diagonale teilen. [1]

Folgende Algorithmen sollen implementiert werden:

- **Ein „effizienterer“ Brute-Force-Algorithmus**
  - In jeder Reihe wird nur eine Dame gesetzt.
  - Es werden alle Möglichkeiten je eine Dame in einer Reihe zu setzen geprüft
  - Korrekte Belegungen werden ausgegeben
- **Backtracking**
  Man setzt die erste Dame und probiert systematisch alle Möglichkeiten für die zweite. Hat man eine Möglichkeit gefunden, sucht man einen Platz für die dritte. Falls man einmal keinen gültigen Platz für die i+1te Dame findet, geht man wieder einen Schritt zurück und versucht die vorhergehende Dame anders zu platzieren, das ist dann der Backtracking-Schritt.

Aufgabe 1

a) Implementieren Sie die Klasse `Chessboard`, welche das Spielbrett darstellt auf dem die Damen gesetzt sind. Es soll möglich sein beliebige n x n Spielbretter zu erzeugen. Ein `Chessboard`-Objekt besteht aus seiner Größe und der gesetzten Damen.

b) Implementieren Sie die Methode `void set(boolean value, int x, int y)`. Diese soll den Wert des entsprechenden Feldes auf dem Spielbrett ändern.

[1] http://de.wikipedia.org/wiki/Damenproblem

Abgabe der Lösung bis zum **22.4.2014, 03:00 Uhr (sehr früh!)** per eMail. Die eMail-Adresse für Ihre Abgabe ist die ihres Tutors. Im StudIP unter Dateien->E-Mail_Tutoren.pdf finden Sie den für ihre Gruppe zuständigen Tutor.

Hängen Sie die .java Dateien mit Ordnerstruktur bitte als .zip Archiv an. **Versenden Sie keine .class Dateien! Dropbox Abgaben o.ä. sind unzulässig!** Bei Mehrfacheinsendungen wird nur die letzte eingesandte Lösung berücksichtigt. Gruppenabgaben sind nicht zulässig. Einsendungen nach dem genannten Termin gelten als nicht bestanden. Es wird voraussichtlich 5 Übungen geben. Voraussetzung für die Zulassung zur Zwischenprüfung ist das Bestehen von mindestens 4 Übungen.

Vorlesung:
Priv.-Doz. Dr.-Ing. Matthias Becker
eMail: xmb@sim.uni-hannover.de

Übungsbetrieb:
B.Sc. Christian Kater
eMail: kater@sim.uni-hannover.de

Aufgabenblatt vom 15.04.2014
Abgabe bis **22.04.2014, 03:00**

c) Implementieren Sie die Methode `boolean get(int x, int y)`. Diese soll den Wert des entsprechenden Feldes zurück.

d) Implementieren Sie die Methode `Chessboard copy()`. Diese soll eine neues `Chessboard`-Objekt erzeugen und die gesetzten Damen und die Dimension in dieses kopieren.

e) Implementieren Sie die Methode `String toString()`. Diese soll ein Spielfeld in Folgender Form als String zurückgeben ausgeben:

- `[ ]` für je ein Feld ohne Dame
- `[X]` für je ein Feld mit Dame

Beispielausgabe (Auf Konsole) für die obige Lösung (s. S. 1):

```
[ ][ ][ ][ ][ ][ ][ ][X]
[ ][ ][X][ ][ ][ ][ ][ ]
[X][ ][ ][ ][ ][ ][ ][ ]
[ ][X][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][X][ ][ ]
[ ][X][ ][ ][ ][ ][ ][ ]
[ ][ ][ ][ ][ ][ ][X][ ]
[ ][ ][ ][ ][X][ ][ ][ ]
```

Hinweis: Sie können Zeilenumbrüche durch den String `\n` bewirken. Der String `hello\nworld` wird auf der Konsole als :

```
hello
world
```

ausgegeben.

Aufgabe 2

a) Implementieren Sie die Abstrakte Klasse `BasicQueenSolver`. `BasicQueenSolver` ist eine gemeinsame Basisklasse für Algorithmen zum lösen des Damenproblems. Dieses soll die folgende abstrakte Methode beinhalten:

- `public abstract void solve()`

b) Implementieren Sie die Methode `boolean checkChessboard(Chessboard board)`, die überprüft ob in jeder Reihe / Spalte / Diagonale nur eine Dame zu finden ist.

Abgabe der Lösung bis zum **22.4.2014, 03:00 Uhr (sehr früh!)** per eMail. Die eMail-Adresse für Ihre Abgabe ist die ihres Tutors. Im StudIP unter Dateien->E-Mail_Tutoren.pdf finden Sie den für ihre Gruppe zuständigen Tutor.

Hängen Sie die .java Dateien mit Ordnerstruktur bitte als .zip Archiv an. **Versenden Sie keine .class Dateien! Dropbox Abgaben o.ä. sind unzulässig!** Bei Mehrfacheinsendungen wird nur die letzte eingesandte Lösung berücksichtigt. Gruppenabgaben sind nicht zulässig. Einsendungen nach dem genannten Termin gelten als nicht bestanden. Es wird voraussichtlich 5 Übungen geben. Voraussetzung für die Zulassung zur Zwischenprüfung ist das Bestehen von mindestens 4 Übungen.

Vorlesung:
Priv.-Doz. Dr.-Ing. Matthias Becker
eMail: xmb@sim.uni-hannover.de

Übungsbetrieb:
B.Sc. Christian Kater
eMail: kater@sim.uni-hannover.de

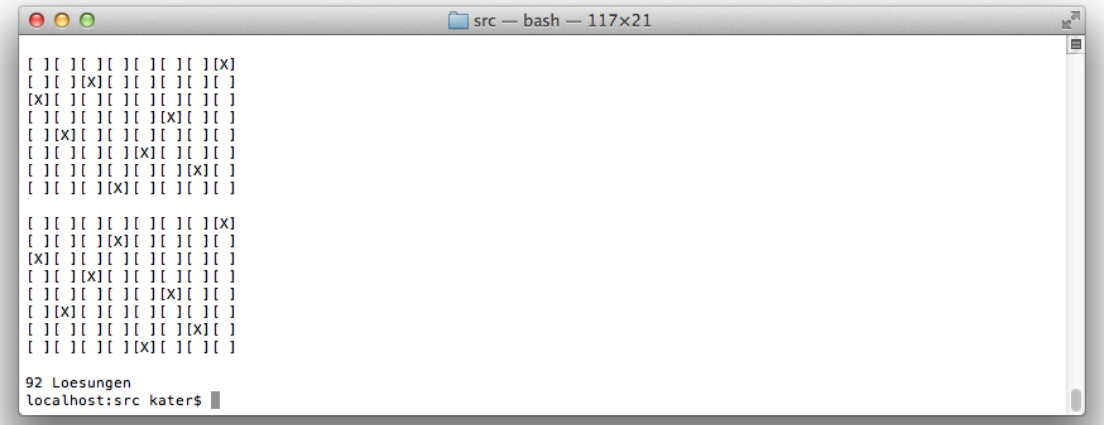
Aufgabenblatt vom 15.04.2014
Abgabe bis **22.04.2014, 03:00**

Aufgabe 3

a) Implementieren Sie die Klasse `BruteForce`, welche von der Klasse `BasicQueenSolver` erbt. Diese soll mittels eines „effizienteren“ BruteForce-Algorithmus die möglichen Lösungen für ein 8x8-Feld berechnen und ausgeben. Beachten Sie die Hinweise zu beginn des Übungsblatts.

b) Implementieren Sie die abstrakte Methode `solve`. Erzeugen Sie alle Kombinationen durch geeignete verschachtelte For-Schleifen. Überprüfen Sie jede Kombination mit `checkChessboard` und geben Sie die akzeptierten Lösungen auf der Konsole aus. Geben Sie zudem nach allen Lösungen die Anzahl der Lösungen aus.

c) Erstellen Sie die Klasse `Main`. Erstellen Sie in dieser eine `Main-Methode` und testen Sie den so eben implementierten Algorithmus.
Hinweis: Für ein 8x8 Feld könnte die Lösung wie folgt aussehen:



Aufgabe 4

a) Implementieren Sie die Klasse `Backtracking`, welche von der Klasse `BasicQueenSolver` erbt. Diese soll alle Lösungen des Damenproblems für ein 8x8-Feld mittels Backtracking ausgeben. Beachten Sie die Hinweise auf der ersten Seite.

b) Implementieren Sie die abstrakte Methode `solve`. Implementieren Sie hierzu den auf der letzten Seite angegebenen Algorithmus.

c) Erweitern Sie die Klasse `Main` so, dass nach dem Brute-Force-Algorithmus der Backtracking-Algorithmus ausgeführt wird.

Abgabe der Lösung bis zum **22.4.2014, 03:00 Uhr (sehr früh!)** per eMail. Die eMail-Adresse für Ihre Abgabe ist die ihres Tutors. Im StudIP unter Dateien->E-Mail_Tutoren.pdf finden Sie den für ihre Gruppe zuständigen Tutor.

Hängen Sie die .java Dateien mit Ordnerstruktur bitte als .zip Archiv an. **Versenden Sie keine .class Dateien! Dropbox Abgaben o.ä. sind unzulässig!** Bei Mehrfacheinsendungen wird nur die letzte eingesandte Lösung berücksichtigt. Gruppenabgaben sind nicht zulässig. Einsendungen nach dem genannten Termin gelten als nicht bestanden. Es wird voraussichtlich 5 Übungen geben. Voraussetzung für die Zulassung zur Zwischenprüfung ist das Bestehen von mindestens 4 Übungen.

Vorlesung:
Priv.-Doz. Dr.-Ing. Matthias Becker
eMail: xmb@sim.uni-hannover.de

Übungsbetrieb:
B.Sc. Christian Kater
eMail: kater@sim.uni-hannover.de

Aufgabenblatt vom 15.04.2014
Abgabe bis **22.04.2014, 03:00**

Pseudocode Backtracking für ein dim x dim-Feld:

static int ***numberOfSolutions***

Solve():

1. ***numberOfSolutions*** = 0
2. Für jede Spalte **i** von 0 bis dim:
    a. Erzeuge ein neues Chessboard und markieren das Feld (0,**i**) mit einer Dame
    b. Rufe **backtracking(1, dim, board)** auf
3. Gebe ***numberOfSulutions*** aus

Backtracking(int n, int dim, Chessboard board)

1. Falls **n == dim**
    a. Gebe **board** aus
    b. ***numberOfSolutions***++
2. Für jede Spalte **i** von 0 bis dim:
    a. Kopiere **board** zu **copy** und markieren das Feld (**n,i**) von **copy** mit einer Dame
    b. Falls **checkChessboard(copy)**
        i. Rufe **backtracking(n+1, dim, copy)** auf

Abgabe der Lösung bis zum **22.4.2014, 03:00 Uhr (sehr früh!)** per eMail. Die eMail-Adresse für Ihre Abgabe ist die ihres Tutors. Im StudIP unter Dateien->E-Mail_Tutoren.pdf finden Sie den für ihre Gruppe zuständigen Tutor.

Hängen Sie die .java Dateien mit Ordnerstruktur bitte als .zip Archiv an. **Versenden Sie keine .class Dateien! Dropbox Abgaben o.ä. sind unzulässig!** Bei Mehrfacheinsendungen wird nur die letzte eingesandte Lösung berücksichtigt. Gruppenabgaben sind nicht zulässig. Einsendungen nach dem genannten Termin gelten als nicht bestanden. Es wird voraussichtlich 5 Übungen geben. Voraussetzung für die Zulassung zur Zwischenprüfung ist das Bestehen von mindestens 4 Übungen.