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 22.04.2014
Abgabe bis **29.04.2014, 03:00**

2. Hausübung

Algorithmen, Packages, JavaDoc, API

In der Software-Entwicklung wird ihr Quellcode von einem anderen Entwickler gewartet oder weiterentwickelt. Daher muss der Quellcode unter anderem gut strukturiert und kommentiert sein, sodass sich andere Entwickler schnell in diesen einarbeiten können. Ein Sprichwort zu diesem Thema, an das Sie bei der Programmierung denken sollten, lautet:

„Programmieren Sie immer so, als wäre der Typ, der den Code pflegen muss, ein gewaltbereiter Psychopath, der weiß, wo Sie wohnen.“

In der ersten Aufgabe sollen Sie ihren Quellcode von letzter Woche aufbessern. Falls Sie das letzte Übungsblatt nicht bearbeitet haben, finden Sie unter Downloads eine Version, die Sie als Grundlage nehmen können.

In den weiteren Aufgaben sollen Sie das Damenproblem mittels Heuristischer Optimierung lösen. Heuristische Optimierung in Bezug auf das Damenproblem können Sie sich wie folgt vorstellen:

- Finden einer Lösung durch intelligentes Probieren, in etwa so, wie ein Mensch vorgehen würde.
    - Zu Beginn wird ein leeres Spielbrett mit Damen gefüllt. Es wird je Zeile die Spalte mit den wenigsten Konflikten zu den vorherigen Damen genommen (Bei Gleichstand entscheidet der Zufall unter den Minima).
    - Anschließend werden solange Damen versetzt, bis eine Lösung gefunden wurde.
    - In jedem Durchlauf:
        - Suche Dame mit den meisten Konflikten
        - Prüfe ob Dame schon im letzten Durchlauf versetzt wurde. Falls ja, nehme eine zufällige Dame.
        - Finde Spalte mit den wenigsten Konflikten für die Zeile der zuvor gefundenen Dame (Bei Gleichstand entscheidet der Zufall unter den Minima).
        - Die Dame wird umgesetzt.

Abgabe der Lösung bis zum **29.04.2014, 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.

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 22.04.2014
Abgabe bis **29.04.2014, 03:00**

Aufgabe 1

a) Unterteilen Sie ihren Quellcode in Packages und ordnen Sie in diese die vorhandenen Klassen sinnvoll ein.
b) Erstellen Sie für den bisherigen Quellcode sinnvolle JavaDoc-Kommentare. Kommentieren Sie zudem kompliziertere Ausschnitte ihres Quellcodes mit einfachen Java-Kommentaren, sodass ihr Quellcode auch für andere verständlich ist. Erstellen Sie auch für die nachfolgenden Aufgaben Java- und JavaDoc-Kommentare.

Aufgabe 2

a) Implementieren Sie die Klasse `Queen`. Ein Queen-Objekt hat eine X- und Y-Koordinate, sowie die Anzahl der in Konflikt stehenden anderen Damen.

b) Lassen Sie die Klasse `Queen` das Interface Comparable implementieren. Es soll ein Queen-Objekt mit einem Anderen anhand der Anzahl der Konflikte verglichen werden.

Hinweis: Schlagen Sie Comparable in der Java-API[1] nach.

Aufgabe 3

a) Implementieren Sie die Klasse `Heuristic`, welche von der Klasse `BasicQueenSolver` erbt. Diese soll **eine** Lösung des Damenproblems für ein 50x50-Feld mittels Heuristischer Optimierung ausgeben.

b) Implementieren Sie die abstrakte Methode `solve`. Implementieren Sie die Heuristische Optimierung wie auf der ersten Seite beschrieben. Falls Sie Probleme haben, orientieren Sie sich an dem Algorithmus und den Hilfsmethoden, die am Ende des Übungsblatts angegeben sind.

Aufgabe 4

a) Packen Sie Ihre gesamte Lösung in ein lauffähiges .jar File. Erstellen Sie zu diesem Zweck eine passende Manifestdatei.

[1] http://docs.oracle.com/javase/7/docs/api/

Abgabe der Lösung bis zum **29.04.2014, 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 22.04.2014
Abgabe bis **29.04.2014, 03:00**

Pseudocode

Lösen des Damenproblems mittels Heuristischer Optimierung:

- Zu Beginn wird ein leeres Spielbrett (Chessboard) mit Größe dim erzeugt
- queens = Leere Liste von Damen (Queen)
- Für jede Zeile i von 0 bis dim
  - Finde die Spalte j in Zeile i mit den wenigsten Konflikten (findBestPosition) zu den bereits gesetzten Damen
  - Setze Dame auf das Feld (i, j)
  - Füge die Dame queens hinzu
- prev = -1
- Solange noch keine Lösung gefunden wurde
  - max = Dame aus queens mit den meisten Konflikten (getMaxima)
  - Falls max in Zeile prev ist
    - max = Zufällige Dame aus queens
  - entferne max aus queens
  - Entferne Dame vom Feld (max.x, max.y)
  - max = Dame in der Spalte mit den wenigsten Konflikten (findBestPosition)
  - Setze Dame auf das Feld (max.x, max.y)
  - Füge max zu queens hinzu
  - prev = max.x
- Gebe Spielbrett aus

Hilfsmethoden

getMaxima(List<Queen> **queens**)

1. Sortiere **queens** aufsteigend
   Hinweis: Listen können mit der Klasse Collection sortiert werden.
2. n = „Wie oft kommt das Maximum von queens in queens vor“
   Hinweis: Nutzen Sie die Sortierung aus (wann unterscheiden sich die letzten Elemente)
3. Gebe **queens**.remove(**queens**.größe - (1 + Zufall[0,n)) zurück

Abgabe der Lösung bis zum **29.04.2014, 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 22.04.2014
Abgabe bis **29.04.2014, 03:00**

findBestPosition(Chessboard board, int x)

1. Erzeuge Liste von Damen
2. minimum = unendlich
3. Überprüfe jede Spalte i
    1. Falls mit einer Dame in Spalte i genauso viele Konflikte wie minimum auftreten
        1. Füge die Dame der Liste hinzu.
    2. Falls mit einer Dame in Spalte i weniger als minimum Konflikte auftreten:
        1. Leere die Liste.
        2. Füge die Dame der Liste hinzu.
        3. Setze minimum auf neues Minimum.
4. Gebe eine zufällige Dame der Liste zurück.

getConflicts(Chessboard board, int x, int y)

1. conflicts = 0
2. Erhöhe conflicts für jede andere Dame in Zeile x um 1
3. Erhöhe conflicts für jede andere Dame in Spalte y um 1
4. Erhöhe conflicts für jede andere Dame in Diagonale von oben links nach unten rechts um 1.
5. Erhöhe conflicts für jede andere Dame in Diagonale von unten links nach oben rechts um 1.
6. Gebe conflicts zurück

Abgabe der Lösung bis zum **29.04.2014, 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.