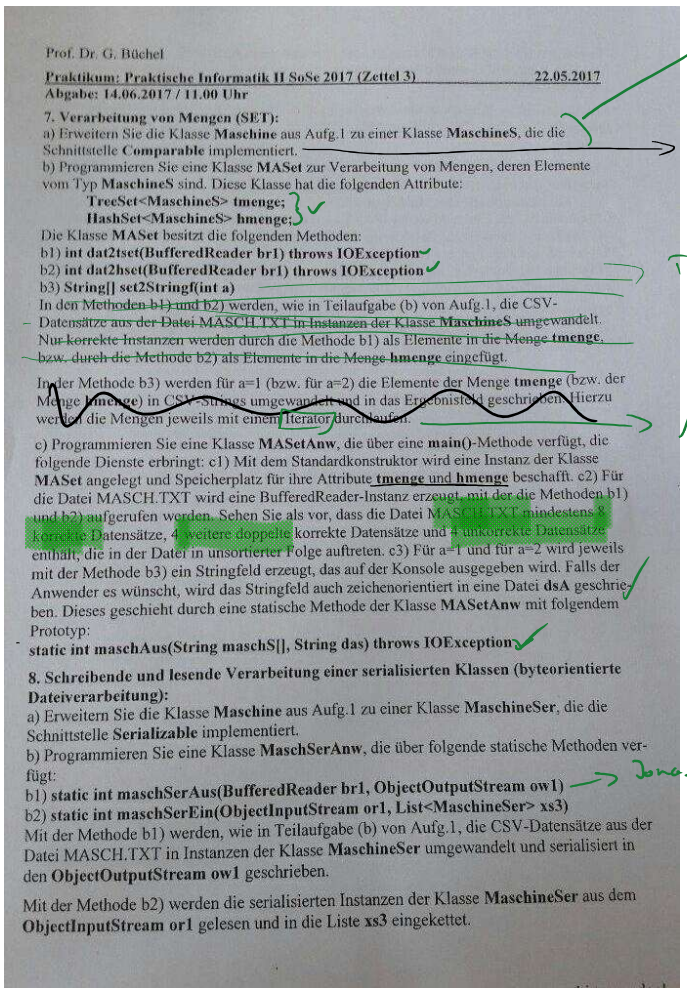
PI2 Versuch 3

Freitag, 26. Mai 2017 18:20



Prof. Dr. G. Büchel

Praktikum: Praktische Informatik II SoSe 2017 (Zettel 3) 22.05.2017
Abgabe: 14.06.2017 / 11.00 Uhr

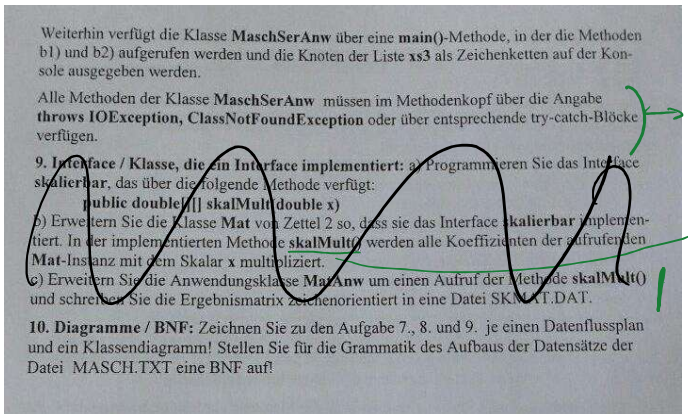
7. Verarbeitung von Mengen (SET):
a) Erweitern Sie die Klasse **Maschine** aus Aufg.1 zu einer Klasse **MaschineS**, die die Schnittstelle **Comparable** implementiert.
b) Programmieren Sie eine Klasse **MASet** zur Verarbeitung von Mengen, deren Elemente vom Typ **MaschineS** sind. Diese Klasse hat die folgenden Attribute:
TreeSet<MaschineS> tmenge;
HashSet<MaschineS> hmenge;
Die Klasse **MASet** besitzt die folgenden Methoden:
b1) **int dat2tset(BufferedReader br1) throws IOException**
b2) **int dat2hset(BufferedReader br1) throws IOException**
b3) **String[] set2Stringf(int a)**
In den Methoden b1) und b2) werden, wie in Teilaufgabe (b) von Aufg.1, die CSV-Datensätze aus der Datei MASCH.TXT in Instanzen der Klasse **MaschineS** umgewandelt. Nur korrekte Instanzen werden durch die Methode b1) als Elemente in die Menge **tmenge**, bzw. durch die Methode b2) als Elemente in die Menge **hmenge** eingefügt.

In der Methode b3) werden für a=1 (bzw. für a=2) die Elemente der Menge **tmenge** (bzw. der Menge **hmenge**) in CSV-Strings umgewandelt und in das Ergebnisfeld geschrieben. Hierzu werden die Mengen jeweils mit einem Iterator durchlaufen.

c) Programmieren Sie eine Klasse **MASetAnw**, die über eine **main()**-Methode verfügt, die folgende Dienste erbringt: c1) Mit dem Standardkonstruktor wird eine Instanz der Klasse **MASet** angelegt und Speicherplatz für ihre Attribute **tmenge** und **hmenge** beschafft. c2) Für die Datei MASCH.TXT wird eine BufferedReader-Instanz erzeugt, mit der die Methoden b1) und b2) aufgerufen werden. Sehen Sie als vor, dass die Datei MASCH.TXT mindestens 8 korrekte Datensätze, 4 weitere doppelte korrekte Datensätze und 5 unkorrekte Datensätze enthält, die in der Datei in unsortierter Folge auftreten. c3) Für a=1 und für a=2 wird jeweils mit der Methode b3) ein Stringfeld erzeugt, das auf der Konsole ausgegeben wird. Falls der Anwender es wünscht, wird das Stringfeld auch zeichenorientiert in eine Datei **dsA** geschrieben. Dieses geschieht durch eine statische Methode der Klasse **MASetAnw** mit folgendem Prototyp:
static int maschAus(String maschS[], String das) throws IOException

8. Schreibende und lesende Verarbeitung einer serialisierten Klassen (byteorientierte Dateiverarbeitung):
a) Erweitern Sie die Klasse **Maschine** aus Aufg.1 zu einer Klasse **MaschineSer**, die die Schnittstelle **Serializable** implementiert.
b) Programmieren Sie eine Klasse **MaschSerAnw**, die über folgende statische Methoden verfügt:
b1) **static int maschSerAus(BufferedReader br1, ObjectOutputStream ow1)**
b2) **static int maschSerEin(ObjectInputStream or1, List<MaschineSer> xs3)**
Mit der Methode b1) werden, wie in Teilaufgabe (b) von Aufg.1, die CSV-Datensätze aus der Datei MASCH.TXT in Instanzen der Klasse **MaschineSer** umgewandelt und serialisiert in den **ObjectOutputStream ow1** geschrieben.

Mit der Methode b2) werden die serialisierten Instanzen der Klasse **MaschineSer** aus dem **ObjectInputStream or1** gelesen und in die Liste **xs3** eingekettet.



Weiterhin verfügt die Klasse **MaschSerAnw** über eine **main()**-Methode, in der die Methoden b1) und b2) aufgerufen werden und die Knoten der Liste **xs3** als Zeichenketten auf der Konsole ausgegeben werden.

Alle Methoden der Klasse **MaschSerAnw** müssen im Methodenkopf über die Angabe **throws IOException, ClassNotFoundException** oder über entsprechende try-catch-Blöcke verfügen.

9. Interface / Klasse, die ein Interface implementiert: a) Programmieren Sie das Interface **skalierbar**, das über die folgende Methode verfügt:
public double[][] skalMult(double x)
b) Erweitern Sie die Klasse **Mat** von Zettel 2 so, dass sie das Interface **skalierbar** implementiert. In der implementierten Methode **skalMult()** werden alle Koeffizienten der aufrufenden **Mat**-Instanz mit dem Skalar **x** multipliziert.
c) Erweitern Sie die Anwendungsklasse **MatAnw** um einen Aufruf der Methode **skalMult()** und schreiben Sie die Ergebnismatrix zeichenorientiert in eine Datei SKMAT.DAT.

10. Diagramme / BNF: Zeichnen Sie zu den Aufgabe 7., 8. und 9. je einen Datenflussplan und ein Klassendiagramm! Stellen Sie für die Grammatik des Aufbaus der Datensätze der Datei MASCH.TXT eine BNF auf!