Name: _____ Vorname:

Matrikelnummer:_____ Testat:_____

8. Praktikum "Algorithmen und Programmierung II"
SS 2013

Vorbemerkungen. Auf der Web-Seite `http://www.gm.fh-koeln.de/ehses/ap` finden Sie alle nötigen Hilfsdateien.

Ziel der folgenden Aufgaben ist es, einige Merkmale der Objektorientierung in Java zu wiederholen (Abstrakte Klasse, geschachtelte Klasse, anonyme Klasse) und Algorithmen zur verketteten Liste zu üben. Gleichzeitig sollen Sie einen Einblick in die Arbeitsweise einer einfachen virtuellen Maschine bekommen.

Die Eingaben für das Testprogramm `Calculator` sind einfache Ausdrücke wie 2.3 * (5 – 0.3)

Aufgabe 1) (Thema Abstrakte Klassen) Korrigieren Sie die als fehlerhaft markierten Klassen des Paketes `tree`.

Aufgabe 2) Vervollständigen Sie die Klasse `util.LinkedObjects`. Die Klasse soll eine zirkuläre doppelt verkettete Liste implementieren. Ergänzen Sie die markierten Methoden,. so dass alle JUnit-Tests im Quelltextfolder `test` erfüllt sind. **Alle Methoden sollen verständlich mit JavaDoc kommentiert werden!**

Für die Vervollständigung der Iterator-Klasse (wo ist diese?) sollen Sie neben den Vorlesungsunterlagen auch die Dokumentation der Java-API (`java.util.Iterator`) durchlesen!

Ergänzen Sie in `main.Calculator.main` die Ausgabe des erzeugten Programms. Testen Sie mit unterschiedlichen Eingaben. Sie sollen den erzeugten Code per Hand interpretieren können.

Aufgabe 3) Ergänzen Sie die fehlenden Teile in der Funktion `vm.VM.compute`.

Am Ende muss alles funktionieren.. Beantworten Sie auch die folgenden Fragen:

Frage 1: Welche Vorteile hat eine zirkuläre doppelt verkettete Liste?

Frage 2: Wo werden die Objekte der Klassen `tree.PlusNode` etc. erzeugt?

Frage 3: Erzeugen Sie für das folgende Java-Programm mittels dem (Kommandozeilen-) Befehl `javap -c` *KlassenName* ein disassembliertes Listing. Vergleichen Sie dieses mit der entsprechenden Befehlsliste der Praktikumsaufgabe.

```
public class Example {
    public static double fkt(double a, double b, double c) {
        return a * b * c;
   }
}
```

Wenn man das Beispiel als Zahlenausdruck (2 * 3 + 4) hinschreibt, berechnet der Java-Compiler sofort das Endergebnis. Wenn die Praktikumsklasse so verfahren wäre, hätten Sie nichts zu tun gehabt.

Anregung für Interessierte: Sie können das auch ganz einfach hinkriegen, wenn Sie eine Klasse schreiben, die die Schnittstelle `parser.Builder<Double>` sinnvoll (wie?) implementiert. Wenn Sie der Funktion `Parser.parse` ein Objekt dieser Klasse mitgeben, erhalten Sie direkt den Ergebniswert. Näheres dazu auch in Praktikum 9.