Prof. Dr. V. Linnemann
Stefan Werner
Universität zu Lübeck
Institut für Informationssysteme

Lübeck, den 12. April 2012

Algorithmen und Datenstrukturen

Sommersemester 2012

2. Übungsblatt

Aufgabe 1: Java Generics

Diese Aufgabe basiert auf der in der Vorlesung vorgestellten generischen Implementierung einer sortierten verketteten Liste (vgl. Abschnitt 0.3). Durch diese allgemeine Implementierung ist es möglich diese Liste auch für andere Objekttypen zu verwenden, **ohne** die Implementierung der Liste ändern zu müssen.

a) Implementieren Sie eine Klasse `BeerBottle` und modellieren Sie darin die Bezeichnung, Brauerei, Alkoholgehalt sowie die Füllmenge. Geben Sie einen Konstruktor an und überschreiben Sie die `toString()`-Methode geeignet. (2 Punkte)

b) Erweitern Sie `BeerBottle`, so dass unter Verwendung des Interface `MyList` und der Klasse `MyLinkedList` (in **unveränderter** Form(!)) eine Liste von Objekten dieses Typs erzeugt werden kann. Die Sortierung soll anhand der Füllhöhe (aufsteigend) realisiert werden. Sollten zwei Flaschen die selbe Füllmenge besitzen, so soll alphabetisch nach der Bezeichnung sortiert werden. Testen Sie Ihre Implementierung in einer `main`-Methode. (7 Punkte)

Aufgabe 2: Rekursive Sinus-Approximation

In dieser Aufgabe soll die Sinus-Funktion unter Verwendung der Bildungsvorschrift

$$sin(x) = \begin{cases} x & \text{falls x } \textit{ausreichend klein} \\ 3 \cdot sin(\frac{x}{3}) - 4 \cdot sin^3(\frac{x}{3}) & \text{sonst} \end{cases}$$

approximiert werden.

Implementieren Sie die **rekursive** Methode `public static double sinRek(double x)`, welche als Parameter den Winkel im Bogenmaß erhält und das Resultat nach der gegebenen Vorschrift berechnet.

Als *ausreichend klein* können in dieser Aufgabe Werte für $x \leq 0.01$ betrachtet werden. Deklarieren Sie diesen Grenzwert zur Vermeidung von *Magic Numbers*[1] als Konstante. Führen Sie eine zusätzliche Variable ein, um die Anzahl der rekursiven Funktionsaufrufe zählen zu können.

Testen Sie Ihre Implementierung mit $x \in \{0, 0.5, \frac{3}{4}\pi, \pi, 2\pi\}$ (die Klasse `Math` der Java-API stellt π als Konstante zur Verfügung). Vergleichen Sie die Genauigkeit mit der API-eigenen Implementierung `Math.sin(double a)` des Sinus. (5 Punkte)

Aufgabe 3: Das Collatz-Problem

Das Collatz-Problem ist ein bis dato ungelöstes mathematisches Problem des Mathematikers Lothar Collatz. Dabei werden Zahlenfolgen nach der folgenden einfachen Bildungsvorschrift konstruiert:

$$x_{n+1} = \begin{cases} \frac{x_n}{2} & \text{falls } x_n \text{ gerade} \\ 3 \cdot x_n + 1 & \text{falls } x_n \text{ ungerade} \end{cases}$$

Es wird vermutet, dass jede so generierte Zahlenfolge ausgehend von einem Startwert $x_0 \in \mathbb{N}\backslash\{0\}$ immer im Zyklus $4, 2, 1, ...$ mündet.

a) Da ein Gegenbeispiel als Gegenbeweis bereits ausreichend ist, sollen diese Zahlenfolgen mittels einer **rekursiven** Methode generiert werden. Überlegen Sie sich eine geeignete Abbruchbedingung, da die oben angegebene Bildungsvorschrift prinzipiell nicht terminiert. (4 Punkte)

b) Geben Sie den Aufrufbaum für $x_0 = 6$ an. (3 Punkte)

c) Implementieren Sie nun eine **iterative** Methode, die die Zahlenfolge generiert. (4 Punkte)

Beachten Sie die Hinweise auf dem ersten Übungsblatt.

Abgabetermin: Donnerstag, den 19. April bis 10 Uhr im Institut für Informationssysteme, 2. OG im Informatik-Neubau (Raum 2.071, Küche neben Sekretariat)

[1] http://de.wikipedia.org/wiki/Magische_Zahl_%28Informatik%29