Objektorientierte Programmierung, Übungsaufgabe 1

Schnittstelle für Langzahl-Arithmetik

Jan Tammen <foobar@fh-konstanz.de>

6. April 2005

Korrekturen und Ergänzungen

Gegenüber dem Original-Konzept vom 29. März 2005 wurden einige Ergänzungen und Korrekturen vorgenommen, welche im Folgenden dokumentiert werden.

1 Schnittstelle der Langzahl-Klasse

Einige Operatoren (`+, -, *`) der Klasse wurden angepasst, sodass sie nicht mehr das Objekt selbst verändern, sondern ein neues `MyNumber`-Objekt für das Ergebnis erzeugen und dieses zurückliefern.

2 Anwendung: Reihenentwicklung

Um bei der Reihenentwicklung alle implementierten arithmetischen Operatoren einzusetzen, werden jetzt die folgenden Reihen verwendet:

$$\sum_{k=1}^{\infty} \frac{1}{k(k+1)} = \frac{1}{2} + \frac{1}{6} + \frac{1}{12} + \ldots = \mathbf{1} \tag{1}$$

$$\sum_{k=1}^{\infty} (-1)^{k+1} \frac{1}{2k-1} = \pm \ldots = \frac{\mathfrak{ß}}{\mathbf{4}} \tag{2}$$

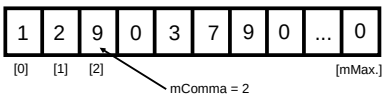
3 Klasse mit Langform

Bei der Implementierung der **Multiplikation** wurde eine Schwäche des zuvor gewählten Ansatzes zur Speicherung der Langzahl deutlich: durch die Trennung von Vor- und Nachkommateil fällt diese Operation recht schwer. Daher wird die Zahl nun *komplett in einem* Feld gespeichert, wodurch u.a. die Multiplikation deutlich leichter zu implementieren ist. Die Position des Kommas wird – relativ zum Beginn des Feldes – in einer `integer`-Variablen gespeichert.

Dadurch ergaben sich natürlich auch Änderungen bei den anderen arithmetischen Operatoren, welche sich aber meiner Meinung nach aus dem entsprechenden Quellcode erschließen.

3.1 Skizze Speicherformat

Graphisch lässt sich eine Langzahl (z.B. `12.90375`) nun folgendermaßen veranschaulichen:



Feld mNumber

4 Addition und Subtraktion

Die in der Dokumentation aufgelisteten Sonderfälle bei Addition und Subtraktion lassen sich jeweils zu einer einzigen Abfrage zusammenfassen.