Technische Universität Berlin **WS 2012/2013**
Fakultät II, Institut für Mathematik
Sekretariat MA 5–2, Frau Kiefer

Dr. Britta Peis
Martin Groß
Dr. Max Klimm
Madeleine Theile

Aylin Acikel, Katharina Bütow, Christian Döblin, Alexander Hopp, Daniel Kuske, Olivia Röhrig, Robert Rudow, Daniel Schmand, Hendrik Schrezenmaier, Judith Simon, Sebastian Spies, Steffen Suerbier, Fabian Wegscheider, Jan Zur

Computerorientierte Mathematik I

Stellenwertsysteme

4. Programmieraufgabe

Abnahme: spätestens am 22./23.11.2012 (je nach Gruppennummer).

Gerade Gruppennummern geben spätestens am Donnerstag, ungerade Gruppennummern geben spätestens am Freitag ab.

Es soll ein Programm erstellt werden, das Zahlen zwischen beliebigen Zahlensystemen umformen kann. Wir alle kennen Zahlen im *Zehner- oder Dezimalsystem*, das heißt jede Ziffer wird (abhängig von ihrer Stelle) mit einer Zehnerpotenz multipliziert, zum Beispiel

$$1034 = 1 \cdot 1000 + 0 \cdot 100 + 3 \cdot 10 + 4 \cdot 1 = 1 \cdot 10^3 + 0 \cdot 10^2 + 3 \cdot 10^1 + 4 \cdot 10^0 = \sum_{i=1}^{k} z_i \cdot 10^{i-1}$$

für $k = 4$ und $z_1 = 4, z_2 = 3, z_3 = 0, z_4 = 1$.

Tatsächlich gibt es eine solche Darstellung nicht nur zur Basis 10, sondern zu jeder natürlichen Basis ab 2. Genauer: Sei $b \in \mathbb{N} \setminus \{0, 1\}$ eine *Basis* und $z \in \mathbb{N}$ eine natürliche Zahl. Dann gibt es ein $n \in \mathbb{N}$ mit $0 \leq z \leq b^n - 1$ und z ist eindeutig darstellbar als

$$\sum_{i=1}^{n} z_i \cdot b^{i-1}$$

und $z_i \in \{0, 1, 2, \ldots, b-1\}$. Wir schreiben eine solche Zahl zur Basis b als $(z_n z_{n-1} z_{n-2} \cdots z_1)_b$ und sprechen von einer *b-adischen* Darstellung. Zum Beispiel erhalten wir für $z = 10$ und $b = 3$

$$10 = 1 \cdot 3^2 + 0 \cdot 3^1 + 1 \cdot 3^0 = (101)_3$$

Besonders wichtig sind die Fälle $b = 2$ (*Dualsystem*) und $b = 16$ (*Hexadezimalsystem*)[1]. Wenn ein Zahlsystem mehr als 10 Ziffern benötigt, werden die auf 9 folgenden Ziffern mit Buchstaben $a, b, c, \ldots$ bezeichnet. Es ist also z.B. $(10)_{10} = (a)_{16}$, $(11)_{10} = (b)_{16}$, $(12)_{10} = (c)_{16}$, $(13)_{10} = (d)_{16}$, $(14)_{10} = (e)_{16}$, $(15)_{10} = (f)_{16}$ und schließlich $(16)_{10} = (10)_{16}$.

Schreibt ein Programm, das einen String `representation` und zwei Basen `basis` und `newBasis` als Eingabe bekommt. Es soll `representation` als Zahl zur Basis `basis` interpretiert werden und als Zahl zur Basis `newBasis` ausgegeben werden. Implementiert dazu die folgenden Methoden.

[1] Weitere Zahlsysteme, die historisch von Interesse sind, sind das *Oktalsystem* zur Basis 8, das *Duodezimalsystem* zur Basis 12 und das *Sexagesimalsystem* zur Basis 60.

(a) Schreibt eine Methode, die ein Zeichen, das entweder eine Ziffer 0-9 oder ein Zeichen a-z enthält, in eine ganze Zahl umwandelt. Dieser Methode wird ein `char` übergeben und der numerische Wert dieses Zeichens wird zurückgegeben. Die Methode soll

```
private static int charToValue( char c )
```

lauten. Die Buchstaben haben als `char` andere Werte als die Zahlen, die sie symbolisieren. So haben die Zeichen "0..9" die Werte 48 bis 57. Die Zeichen "a..z" haben die Werte 97 bis 122. Die Methode `charToValue` soll also die gegebenen Zeichen in die passenden Zahlwerte umwandeln, also z. B. '0' $\mapsto$ 0 oder 'a' $\mapsto$ 10. Es ist nicht erlaubt, die Methode `Charachter.getNumericValue( char )` zu benutzen.

Hinweis: Man kann mit `char`-Variablen rechnen. Es ist also z. B. möglich `int x = c - 48` oder `int x = c - '0'` für einen `char c` zu schreiben.

(b) Zuerst soll eine Zahl vom System mit Basis `basis` in einen `int` umgewandelt werden. Schreibt dazu eine Methode

```
private static int toInt( String representation, int basis )
```

die eine Basis `basis` und einen String `representation` als Eingabe bekommt. Der String soll als Zahl zur Basis `basis` interpretiert werden und anschließend in einen `int` umgewandelt werden.

Hinweis: Nutzt dazu die Methode aus (a).

Der Ausdruck a^b kann mit der Funktion `Math.pow( double a, double b)` berechnet werden.

Der folgende Algorithmus in Pseudocode beschreibt die Umwandlung einer Zahl $z = (z_n z_{n-1} z_{n-2} \cdots z_1)_b$ in eine Dezimalzahl.

toInt(z, b)
Input: $z = (z_n z_{n-1} z_{n-2} \cdots z_1)_b$, $b \in \{2, 3, \ldots\}$ Basis
Output: Wert $\sum_{i=1}^{n} z_i \cdot b^{i-1}$ von z

$v := 0$
FOR $i := 1$ **TO** n **DO**
    $v := v + z_i \cdot b^{i-1}$
ENDFOR
RETURN v

(c) Analog zur Methode aus (a) sollt ihr nun die Methode

```
private static char valueToChar( int v )
```

implementieren. Ihr wird eine Zahl v übergeben und sie soll das passende Symbol zurückgeben. Also z. B. 0 $\mapsto$ '0' oder 10 $\mapsto$ 'a'. m

Hinweis: Java wandelt eine Variable vom Typ `int` nicht automatisch in einen `char` um, dafür wird ein expliziter Cast benötigt.

(d) Implementiert nun eine Methode

```
private static String fromInt( int n, int basis )
```

die eine Basis `basis` und eine Zahl `n` in Dezimaldarstellung als Eingabe bekommt und einen String. Es soll die Zahl `n` zur Basis `basis` als String zurückgegeben werden. **Hinweis:** Der folgende Pseudocode erläutert die Umrechnung.

fromInt(n, b)
Input: n Dezimalzahl, $b \in \{2, 3, \ldots\}$ Basis
Output: String $z_n z_{n-1} z_{n-2} \cdots z_1$, der n zur Basis b darstellt
$s := \emptyset$
DO
  $r := n \bmod b$
  $n := \lfloor n/b \rfloor$
  $s :=$ valueToChar(r) $+s$
WHILE $n > 0$
RETURN s

(e) Lest nun in eurer `main`-Methode die Zahl `representation`, die Basis `basis` und die Zielbasis `newBasis` ein. Überlegt euch, bis zu welcher Basis Zahlen mit euren Methoden umgewandelt werden können. Gebt beim Überschreiten dieser Genzen entsprechende Fehlermeldungen aus. Sorgt außerdem dafür, dass für ungültige Eingaben für die Basis und für negative Dezimalzahlen eine Fehlermeldung angezeigt wird. Nutzt zunächst die Methode aus (b), um die Zahl ins Zehnersystem umzuwandeln. Anschließend ruft die Methode aus (d) auf, um die Zahl aus dem Zehnersystem ins System zur Basis `newBasis` umzuwandeln.

Viel Spaß und Erfolg!