6. Übung **Informatik A** **WS 07/08**

Klaus Kriegel **Abgabe: 04.12.2007, 12:00 Uhr**

Aufgabe 1: Euklidischer Algorithmus (2 + 4 + 2 Punkte)

Aus der Vorlesung kennen wir den Fakt, dass für zwei beliebige ganze Zahlen a und b ganze Zahlen s und t existieren, so dass $ggT(a, b) = s \cdot a + t \cdot b$ erfüllt ist, und eine Methode zur Berechnung von s und t.

a) Wenden Sie diese Methode an, um die Koeffizienten s und t für die Zahlen $a = 130$ und $b = 48$ zu bestimmen.

b) Seien $a, b, c \in \mathbb{Z}$ Zahlen mit der Eigenschaft, dass es keine ganze Zahl $d > 1$ gibt, die alle drei Zahlen teilt. Zeigen Sie, dass dann ganze Zahlen s, t, u existieren, so dass $1 = sa + tb + uc$, und erläutern Sie, wie man diese Werte bestimmen kann.

c) Wenden Sie Ihre in b) beschriebene Methode zur Berechnung von s, t und u für $a = 12, b = 20$ und $c = 75$ an.

Aufgabe 2: Datentyp `Char` (2 + 2 + 4 Punkte)

Eine bei Laien beliebte – für wirklich geheime Nachrichten keinesfalls zu empfehlende – Methode zur Verschlüsselung von Texten basiert auf der Vertauschung von Buchstaben.

a) Implementieren Sie dazu eine Methode `reflectLower :: Char -> Char` , die nur auf Kleinbuchstaben operiert und die Spiegelung der alphabetischen Anordnung realisiert, d.h. `a` wird auf `z` abgebildet, `b` auf `y` usw. Für alle anderen Zeichen soll die Fehlermeldung `"out of range"` ausgegeben werden.

b) Implementieren Sie eine Methode `reflectLetter :: Char -> Char` , die wie folgt auf allen Groß- und Kleinbuchstaben operiert: $a \mapsto Z$, $b \mapsto Y$, $\ldots, z \mapsto A$, $A \mapsto z$, $\ldots, Z \mapsto a$. Für alle anderen Zeichen soll die Fehlermeldung `"out of range"` ausgegeben werden.

c) Mit der dritten Methode `code :: Char -> Char -> Char -> Char` sollen die Ziffern mit kleinen Buchstaben codiert werden. Zusätzlich zu der zu codierenden Ziffer werden zwei kleine Buchstaben nennen wir sie l_0 und l_1 übergeben, die bereits die Ausgaben für die Ziffern `'0'` und `'1'` angeben. Zur Codierung des Nachfolgers einer Ziffer geht man in der alphabetischen Reihenfolge jeweils soviele Schritte weiter, wie durch den Abstand zwischen l_0 und l_1 vorgegeben ist – gelangt man dabei an das Ende des Alphabets, springt man zurück auf den Anfang. Die Methode sollte bei allen ungültigen Eingaben die Fehlermeldung `"out of range"` ausgeben.
Beispiel: Der Aufruf `code '4' 's' 'v'` sollte den Wert `'e'` erzeugen, denn der Abstand zwischen `'s'` und `'v'` ist 3 und folglich wird die `'2'` auf `'y'`, die `'3'` auf `'b'` und die `'4'` auf `'e'` abgebildet.

Aufgabe 3 auf der nächsten Seite

Aufgabe 3: **2–Komplement** (4 + 3 Punkte)

Achtung: Physiker müssen diese Aufgabe nicht bearbeiten !!!

a) Stellen Sie die Zahlen $x = -53$ und $y = -67$ und $z = -79$ als 8–Bit Zahlen im 2–Komplement dar.
Führen Sie die Additionen $x + y$, $x + z$ und $y + z$ aus, wobei ein eventueller Überlauf nach Addition der Vorzeichenbits ignoriert wird. Kommentieren Sie die Ergebnisse.

b) Zeigen Sie, dass die Addition von zwei negativen Zahlen n–Bit Zahlen im 2–Komplement das korrekte Ergebnis liefert, sofern die Summe der zwei Zahlen im darstellbaren Bereich liegt. Sie können eine Begründung nach dem im Skript vorgegebenen Schema entwickeln.