Praktikum Programmieren
Studiengang Angewandte Informatik
Prof. Dr. Bernd Kahlbrandt
Dipl.-Ing. Gerhard Oelker
Hochschule für Angewandte Wissenschaften Hamburg
Department für Informatik
17. Oktober 2019

Aufgabenblatt A03: Einheitenumrechnung

In diesem Aufgabenblatt sollen Sie am Beispiel der Umrechnung von Einheiten eine Reihe von wichtigen Ruby-Konstrukten üben. Den Umgang mit Strings und Zahlen kennen Sie aus den vorangehenden Aufgabenblättern. In diesem Aufgabenblatt werden Sie weitere Datenstrukturen und viele weitere Konstrukte üben können. Welche Ruby-Konstrukte Sie dabei verwenden, überlassen ich Ihnen, aber machen Sie sich das Leben nicht zu schwierig! Sie können z. B. ein (oder mehrere) Ruby-Skripte mit geeigneten Methoden schreiben, eine oder mehrere Klassen.

Es gibt verschieden Einheiten für physikalische Größen: Im MKS System gibt es m, kg, s und die entsprechenden Bruchteile. Im angelsächsischen System gibt es $inch$, $feet$, $mile$.

Sie sollen in dieser Aufgabe eine einfache Version eines Konvertierungsprogramms in Ruby schreiben, das die Umrechnung von Größen in einer unterstützten Einheit in eine andere unterstützte Einheit ermöglicht. Für dieses Aufgabenblatt können Sie sich auf die Umrechnung von Längen, Massen und Temperaturen zwischen folgenden Einheiten beschränken:

Länge mm, cm, m, km, in, ft, yd, mile, potrzebie, mp, Kp, Fp

Masse g, kg, Zentner, Tonne, ounce, ound, ton, ngogn, blintz, Kn

Temperatur Celsius, Kelvin, Fahrenheit, Smurdley

Alte Einheiten Dutzend (12), Mandel (15), Schock (60), Gross (144).

Bei der Aufzählung der Einheiten für Längen, Massen und Temperaturen ist die Reihenfolge metrisches, angelsächsisches, Potrzebie System (zuerst in [Knu57] publiziert).

Ihre Aufgaben:

1. Schreiben Sie bitte Ruby-Code der Folgendes leistet: Umrechnungen zwischen den genannten Einheiten. Achten Sie ggf. darauf, wie genau (d. h. wieviele Nachkommastellen) die Darstellung der Dezimalzahlen ist. Der Code soll leicht für weitere Einheiten erweiterbar sein, z. B. indem die unterstützten Einheiten und Konvertierungsdatenaus einer Datei eingelesen werden, so können etwa Einheiten für weitere Zehnerpotenzen etc. leicht ergänzt werden.

   Bei den Anzahlen sollen ganzzahlige Umrechungen erfolgen, also z. B. 5 Mandeln $\triangleq$ 1 Schock, 1 Mandel oder 5 Mandeln $\triangleq$ 75.

2. Für Ein- und Ausgaben können Sie die Konsole verwenden. Sowohl *puts* als auch *gets* sollten Sie jetzt kennen. Sehen Sie sich ferner ggf. nochmal *chomp* und *chop* an.

3. Es sollen Eingaben von numerischen Werten mit zugehörigen Einheiten und einer Ausgabeeinheit möglich sein.

4. Sie sollen also einen Ausgangswert und eine Ausgangseinheit (z. B. 42 *mm*) sowie einer Zieleinheit(z. B. *inch*) einlesen können und den zugehörigen Wert in der Zieleinheit (mit Einheit) ausgeben. Die Ausgaben sollen geeignet formatiert werden. Tipp: Sehen Sie sich die Methode *printf* an!

5. Achten Sie bitte darauf, das nur geeignete Einheiten in einander umgerechnet werden können, also Länge in Länge, Temperatur in Temperatur etc.

6. Schreiben Sie bitte Testfälle, mit denen Sie Ihre Umrechnungen überprüfen! So werden Sie auch eher kleine, testbare Methoden schreiben als zu große.

Hinweise:

Ich empfehle ganz dringend eines Aufgliederung in mehrere Skripte bzw. Klassen und auf jeden Fall Methoden, etwa:

1. Für die Interaktion mit Benutzer*inn*en (über die Konsole).

2. Für die Umrechnungen.

Zur Überprüfung Ihrer Ergebnisse können Sie z. B. den Windows Rechner, Google Rechner oder Wolfram Alpha heranziehen, dort ist jeweils zumindest ein Teil dieser Systeme implementiert.

Literatur

[Knu57] Donald E. Knuth. The Potrzebie System of Weights and Measures. *MAD*, 1(33):36–37, Juni 1957. Nachgedruckt in [Knu11].

[Knu11] Donald Ervin Knuth. *Selected Papers on Fun & Games*. CSLI, Stanford, 2011, xvii+741 Seiten. Viele interessante Artikel, u. a. der erste, den Don publizierte.

Der Abgabetermin für alle ist:

Montag, 28.10.2018, 12:00