Informatik I: Einführung in die Programmierung

Prof. Dr. Bernhard Nebel
Dr. Christian Becker-Asano, Dr. Stefan Wölfl
Wintersemester 2013/2014

Universität Freiburg
Institut für Informatik

Übungsblatt 2

Abgabe: Freitag, 8. November 2013, 10:00 Uhr

Wichtige Hinweise: Zur Bearbeitung der Übungsaufgaben legen Sie bitte ein neues Unterverzeichnis `sheet02` im Wurzelverzeichnis Ihrer Arbeitskopie des SVN-Repositories an. Ihre Lösungen werden dann entsprechend des ersten Übungsblatts in Dateien in diesem Unterverzeichnis erwartet: die Lösung zur Aufgabe 2.1 also in der Datei `ex02-1.txt`, etc.

Beachten Sie bei allen Aufgaben die Formatierungshinweise, die in Aufgabe 1.6 auf Blatt 1 angegeben wurden. Insbesondere kopieren Sie bitte bei allen Aufgaben, die in der Python-Shell (-Konsole) zu bearbeiten sind, die Lösung (d.h. Ihre Befehle in der Shell einschließlich Python-Prompt `>>>` sowie deren jeweilige Ausgabe) in die entsprechende Datei und rücken Sie den Code um vier Leerzeichen ein. Bei Aufgaben mit Freitextantworten ist das Einrücken nicht erforderlich.

Überprüfen Sie, dass Sie alle Lösungen ins Repository hochgeladen haben (z.B. mit dem Befehl `svn status`). Überprüfen Sie auch die Webseite Ihres SVN-Unterverzeichnisses:

`https://daphne.informatik.uni-freiburg.de/svn/infoI1314/$RZLOGIN`

Bewertet wird bei allen Aufgaben die letzte Version, die zur Deadline des Übungsblattes auf dem SVN-Server eingereicht ist.

Aufgabe 2.1 (Funktionen: Stellenanzahl in Dezimaldarstellung; 4 Punkte)

Definieren Sie in der Python-Shell eine Funktion `declen`, die bei Eingabe einer nicht-negativen Zahl vom Type `int` die Anzahl der Stellen der Zahl in Dezimalschreibweise zurückgibt. Bei anderen Eingaben (negativ oder von einem anderen Typ) soll `None` zurückgegeben werden.

Testen Sie Ihre Funktion an geeigneten Beispielen, also z.B. mit:

```
>>> declen(0) ==  1
True
>>> declen(100) == 3
True
>>> declen(0b10000) == 2
True
```

Aufgabe 2.2 (Funktionen: Celsius nach Fahrenheit; 1.5+4.5)
Bearbeiten Sie die folgenden beiden Teilaufgaben in der Python-Shell.

(a) Definieren Sie eine Funktion `cels_to_fahrenh(x)`, die bei Eingabe eines numerischen Wertes, gelesen als Temparaturwert in Grad Celsius, den entsprechenden Temperaturwert in Grad Fahrenheit als Fließkommazahl zurückgibt (sofern dies für den eingegeben Wert möglich ist).

(b) Definieren Sie eine Funktion `print_ctof_table(x, y, z)`, die eine Tabelle von Temperaturwerten in Grad Celsius (in aufsteigender Reihenfolge) und entsprechenden Werten in Grad Fahrenheit in der Ausgabe erzeugt. Um die Tabelle variabel zu gestalten, soll die Funktion mit drei Argumenten definiert werden, wobei an den Parameter `x` der erste (= kleinste) Celsius-Wert in der Tabelle, an `y` eine obere Schranke für die dargestellten Celsius-Werte, und an `z` die Schrittweite in den Celsius-Werten übergeben wird (alle Werte als *floats*). Falls die Schrittweite nicht positiv oder die obere Schranke kleiner als der erste Wert ist, soll in der Tabelle nur die Umrechnung für den kleinsten Celsius-Wert ausgegeben werden. Alle Zahlenwerte in der ausgegebenen Tabelle sollen auf eine Nachkommastelle gerundet werden. Die ausgegebene Tabelle muss für alle Celsius-Werte $0 \leq x \leq 99$ wohl-formatiert sein (siehe Beispiel).

In Ihrer Lösung dürfen als einzige Funktionen die in 2.2(a) definierte Funktion `cels_to_fahrenh`, die Funktionen `print`, `float` und `round`, sowie arithmetische, boolsche und Vergleichs-Operatoren verwendet werden. Das Gradzeichen kann durch den String (Zeichenkette) `\u00B0` ausgegeben werden. Prüfen Sie Ihre Lösung an geeigneten Beispielen.

```
>>> print_ctof_table(0.0, 99.0, 17.0)
----------------------
 Celsius   Fahrenheit
----------------------
   0.0°C       32.0°F
  17.0°C       62.6°F
  34.0°C       93.2°F
  51.0°C      123.8°F
  68.0°C      154.4°F
  85.0°C      185.0°F
----------------------
>>> print_ctof_table(0.0, 99.0, -13.0)
----------------------
 Celsius   Fahrenheit
----------------------
   0.0°C       32.0°F
----------------------
```

Aufgabe 2.3 (Zustandsdiagramme; 4 Punkte)

Angenommen, Sie hätten in der Python-Konsole die folgende Befehlssequenz eingegeben. Wie sieht das Zustandsdiagramm an den mit (1), (2) und (3) markierten Stellen aus?

```
>>> s = "spam"
>>> s = s * 3
>>> a = s
>>> def foo():
...     a = "ham"
...     s = "egg"
...     return True
...
>>> foo()
>>>                    # (1)
>>> f = foo
>>> x = f()
>>> if x:
...     s = "spam"
... else:
...     a = "egg"
...
>>> a == s
>>>                    # (2)
>>> i = int(x)
>>> while i <= 3:
...     s = s + s
...     i = i + 1
...                    # (3)
```

Aufgabe 2.4 (Komposition von Funktionen; 4 Punkte)

Definieren Sie in der Python-Shell eine Funktion `comp(f, g)` mit zwei Argumenten. Die Funktion erwartet am Parameter `f` eine beliebige arithmetische (Python-) Funktion mit zwei Argumenten und am Parameter `g` eine arithmetische Funktion mit einem Argument. Es wird ferner angenommen, dass beide Funktionen, die an `comp` übergeben werden, einen Wert vom Typ `int` zurückliefern, wenn sie selbst mit Werten vom Typ `int` aufgerufen werden. Der Aufruf der Funktion `comp` auf zwei solche Funktionen soll eine Funktion zurückliefern, die, wenn aufgerufen mit zwei Integer-Werten `x` und `y`, als Ergebnis den Wert von `f(g(x), g(y))` zurückliefert.
Testen Sie Ihre Funktion an geeigneten Beispielen, z.B. wie folgt:

```
>>> def foo(x, y):
...     return x + y
```

```
...
>>> def bar(x):
...     return 2 * x
...
>>> f = comp(foo, bar)
>>> f(1, 2) == 6
True
```

Aufgabe 2.5 (Erfahrungen; 2 Punkte)

Legen Sie im Unterverzeichnis `sheet02` eine Textdatei `erfahrung.txt` an. Notieren Sie in dieser Datei kurz Ihre Erfahrungen beim Bearbeiten der Übungsaufgaben (Probleme, benötigter Zeitaufwand nach Teilaufgabe, Interessantes, etc.).