Informatik I: Einführung in die Programmierung

Prof. Dr. Peter Thiemann
Dr. Daniel Büscher, Hannes Saffrich
Wintersemester 2019

Universität Freiburg
Institut für Informatik

Übungsblatt 1 – Lösungen

Abgabe: Montag, 28.10.2019, 9:00 Uhr morgens

Aufgabe 1.1 (DAPHNE und Subversion; 2 Punkte)

Im Guide zur Vorlesung finden Sie ein Tutorial, welches die Registrierung in DAPHNE sowie die Installation und Verwendung eines Subversion-Clients beschreibt. Lesen Sie das Tutorial vollständig durch. Führen Sie die ersten vier Schritte (Registrierung in DAPHNE, Installation eines Subversion-Clients, Erstellen einer Arbeitskopie, Publizieren von Dateien) aus. Da Lösungen ausschließlich auf elektronischem Wege über das Kursverwaltungssystem DAPHNE eingereicht werden können, sind diese Schritte unbedingt erforderlich.

Aufgabe 1.2 (Algorithmen; Datei: `algorithmen.txt`; 5 Punkte)

Handelt es sich bei der folgenden umgangssprachlichen, prozeduralen Beschreibung um einen Algorithmus? Entscheiden Sie dazu, ob diese Beschreibung die Bedingungen *Präzision*, *Effektivität*, *statische Finitheit*, *dynamische Finitheit* und *Terminierung* (siehe Folien) erfüllt. Begründen Sie jeweils kurz Ihre Antwort.

> *Gegeben seien ganze Zahlen a, b als Eingabe. Setze $k = 0$. Solange a größer ist als b, führe die folgenden Schritte durch: ziehe b von a ab und erhöhe k um 1. Ist a kleiner oder gleich b, gib k aus.*

Ihre Lösung wird in der Datei `algorithmen.txt` im Unterverzeichnis `sheet01` erwartet (siehe Tutorial, Schritt 4). Beachten Sie die Formatierungshinweise zur Abgabe von Lösungen zu Freitextaufgaben (siehe Guide).

Lösung:

- Präzision ist erfüllt: Jeder Schritt hat eine klare, eindeutige Interpretation.
- Effektivität ist erfüllt: Kein Schritt fordert etwas Unmögliches wie z.B. eine ganze Zahl zu finden die größer und kleiner 0 gleichzeitig ist.
- Statische Finitheit ist erfüllt: Die prozedurale Beschreibung ist ein endlicher Text.
- Terminierung ist nicht generell erfüllt: Für Eingaben mit $a > b$ und $b < 0$ wird die Abbruchbedingung $a \leq b$ nie erfüllt.
- Dynamische Finitheit ist nicht erfüllt: Für alle Eingaben a und b wird lediglich Speicher für die Werte von a, b und k benötigt. Da es sich bei den Werten aber um ganze Zahlen handelt, die beliebig groß werden können, kann die dynamische Finitheit dennoch verletzt werden. Bei den Eingaben, die nicht

terminieren, wird b mit jedem Schleifendurchlauf größer, dynamische Finitheit ist also nicht gegeben.

Die prozedurale Beschreibung ist also kein Algorithmus im Sinne der Vorlesung, da Terminierung und dynamische Finitheit nicht erfüllt sind.

Aufgabe 1.3 (Python: Erste Schritte; 6+5 Punkte)
Installieren Sie wie in der Vorlesung angegeben Python 3 (aktuelle Version: 3.7.4).

Erstellen Sie für jede der folgenden Teilaufgaben eine entsprechende `.py` Datei im Unterverzeichnis `sheet01`. Fügen Sie die Dateien Ihrer Arbeitskopie hinzu und publizieren Sie die Dateien auf dem Subversion-Server.

(a) **Datei** `pattern.py`

Verwenden Sie einen einzelnen Aufruf von `print` um folgende Ausgabe zu erzeugen:

```
===---===---===---===---===---
```

In Ihrem Python Code dürfen die Zeichen `'='` und `'-'` aber jeweils nur ein einziges mal vorkommen.

Lösung:

```
print(('=' * 3 + '-' * 3) * 5)
```

(b) **Datei** `numbers.py`

Finden Sie eine Möglichkeit die folgenden Zahlen so mit den Operationen `+`, `-`, `*` und `//` zu verknüpfen, dass jede Zahl genau ein mal vorkommt und das Ergebnis eine Zahl zwischen 550 und 560 ist.

10 6 2 6 3 1

Beispiel: der Ausdruck

$$(10 + 6) * 2 + 3 * 1 - 6$$

hat eine korrekte Form, aber ergibt 29.

Benutzen Sie in Ihrer Python Datei `print` um sich die Berechnung in folgender Form auszugeben:

```
(10 + 6) * 2 + 3 * 1 - 6  =  29
```

Das Ergebnis der Ausgabe soll durch Python-Code berechnet werden. Folgendes ist also keine Lösung:

```
print("(10 + 6) * 2 + 3 * 1 - 6  =  29")
```

Lösung:

```
print(
  "((10 * 6) + 1) * (6 + 3) + 2  = ",
   ((10 * 6) + 1) * (6 + 3) + 2
)
```

Aufgabe 1.4 (Erfahrungen; Datei: `erfahrungen.txt`; Punkte: 2)

Legen Sie im Unterverzeichnis `sheet01` eine Textdatei `erfahrungen.txt` an. Notieren Sie in dieser Datei kurz Ihre Erfahrungen beim Bearbeiten der Übungsaufgaben (Probleme, Bezug zur Vorlesung, Interessantes, benötigter Zeitaufwand, etc.).

Hinweise

Sofern Sie alle Aufgaben bearbeitet haben, sollte das Verzeichnis für Übungsblatt 1 die folgende Struktur aufweisen:

```
sheet01
├── algorithmen.txt
├── erfahrungen.txt
├── numbers.py
└── pattern.py
```

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

1. **Überprüfen Sie, dass Sie alle Lösungen ins Repository hochgeladen haben (z.B. mit dem Befehl `svn status`).**
2. **Überprüfen Sie auch die Webseite Ihres Daphne/SVN-Verzeichnisses.**