Einführung in die Programmierung

Prof. Dr. Peter Thiemann
Marius Weidner, Hannes Saffrich
Simon Dorer, Sebastian Klähn

Universität Freiburg
Institut für Informatik
Wintersemester 2024

Übungsblatt 1

Abgabe: Montag, 21.10.2024, 09:00 Uhr

Wichtig:

Bevor Sie mit den Aufgaben beginnen, melden Sie sich bitte **jetzt** für ein Tutorat auf HisInOne an.

Aufgabe 1.1 (Setup und Abgaberegeln)

In dieser Aufgabe sollen Sie die für die Vorlesung benötigte Software installieren und sich mit unserer Webplattform zur Abgabe der Übungen vertraut machen. Schauen Sie sich hierzu die Anleitung zur Installation an. Eine Zusammenfassung der im Video gezeigten Installationsschritte finden Sie hier.

Sollten Sie bei der Installation Probleme haben, dann können Sie gerne im Chat[1] oder in den Tutorien fragen. Wir sind auch ganz lieb und beißen nicht :)

Zusätzlich findet am Mittwoch, den 16.10 von 12:15 - 13:00 Uhr in Raum HS 00 026 eine Erklärung zum Abgabesystem statt.

Für die Abgabe ihrer Lösungen beachten Sie bitte folgende Regeln:

1. Verwenden Sie nur Befehle und Programmiertechniken, die Inhalt der *bisherigen* Vorlesungen (bis zum Abgabetermin) und Übungsblätter waren.
2. Die einzelnen Aufgaben sind mit vollständigen Dateinamen annotiert. Halten Sie sich an die Namen und die dazugehörigen Dateiformate. Um Tippfehler zu vermeiden, können Sie die Dateinamen auch einfach kopieren.
3. Ihre Dateien sollen immer in *plaintext* sein (UTF-8 codiert). Dabei haben Textdateien die Endungen `.txt` oder `.md` und Dateien mit Python-Code die Endung `.py`. **Insbesondere sind also keine PDFs, keine Word-Dokumente und auch keine Bildschirmfotos erlaubt!**
4. Geben Sie Ihre Aufgaben stets über unser git-System ab. Abgaben per Mail können *nicht* berücksichtigt werden.
5. Nach dem Hochladen ihrer Lösung prüft der Build-Server, ob ihr Code die Style Guidelines von `flake8` [2] einhält. Dieser Check muss erfolgreich durchlaufen. Sie erkennen dies daran, dass auf der Webplattform unseres git-systems ein grüner Haken zu sehen ist (siehe Abbildung 1).

[1] Auf `https://git.laurel.informatik.uni-freiburg.de` mit Ihrem RZ-Account (wie bei Ilias) einloggen und oben im Menü auf `chat` klicken.

[2] Details hierzu: `https://flake8.pycqa.org/en/latest/`

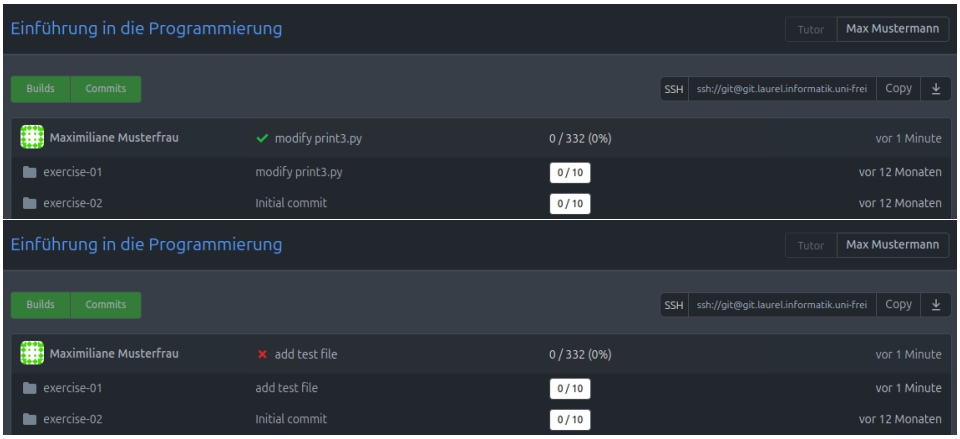

Abbildung 1: Erfolgreicher Build (oben) und fehlerhafter Build (unten)

6. Eine einfache Möglichkeit diese Kriterien zu erfüllen ist es alle Dateien mit Visual Studio Code zu erstellen bzw. zu bearbeiten. Wenn Sie zum Beispiel eine Datei mit `.txt` Endung bearbeiten, dann erkennt Visual Studio Code, dass es sich nicht um ein Python-Skript handelt und verhält sich wie ein normaler Texteditor. Beim Bearbeiten von `.py` Dateien werden `flake8`-Warnungen gelb markiert und mit einem Mouseover erklärt.

Aufgabe 1.2 (Verrückte Division; 3 Punkte; Datei: `division.txt`)

Gegeben sei die folgende umgangssprachliche, prozedurale Beschreibung:

Verrückte Division

1: Gegeben seien zwei ganze Zahlen x und y aus der Menge der ganzen Zahlen $\mathbb{Z}$.
2: Teile x durch y.
3: Ist das Ergebnis ein ganzzahliger Quotient, gib diesen aus und beende das Programm.
4: Falls das Ergebnis kein ganzzahliger Quotient ist erhöhe x um 1 und verringere y um 1.
5: Wiederhole den Vorgang ab Schritt 2.

Handelt es sich bei der Beschreibung um einen Algorithmus? Entscheiden Sie hierfür, ob die Eigenschaften Präzision, Effektivität, statische Finitheit, dynamische Finitheit und Terminierung (siehe Folien) erfüllt sind. Begründen Sie jeweils kurz Ihre Antwort.

Aufgabe 1.3 (Ausdrücke; 4 Punkte; Datei: `expressions.txt`)

In dieser Aufgabe geht es um Unterschiede zwischen dem *Skript-Modus* und dem *interaktiven Modus* von Python.

Im Skript-Modus können sie ein Python-Skript (Dateiendung `.py`) ausführen, indem sie es in Visual Studio Code öffnen und rechts oben auf das ▷-Symbol klicken. Dabei öffnet sich ein Terminal, in dem das Skript ausgeführt wird und die Ausgabe angezeigt wird.

Im interaktiven Modus können Sie Python-Befehle direkt in der Konsole eingeben und erhalten sofort eine Antwort. Öffnen Sie hierzu ein Terminal und geben Sie den Befehl `python3.12` ein. Nun können Sie Python-Code hinter dem Prompt-Zeichen `>>>` eingeben. Beenden können Sie den Modus u.a. mit dem Befehl `exit()`.

(a) (2 Punkte) Wenn Sie nacheinander die Ausdrücke `42` und `print(42)` in die interaktive Konsole eingeben, erhalten Sie die gleiche Ausgabe. Dennoch sind die beiden Ausdrücke nicht identisch. Versuchen Sie die Unterschiede herauszufinden, indem Sie die beiden Ausdrücke auch im Skript-Modus ausführen. Erklären Sie Ihre Erkenntnisse in eigenen Worten.

(b) (2 Punkte) Wenn Sie `print(2 * ((1337 + 1) // 3) + (19 * 42)` in die interaktive Konsole eingeben, sehen Sie statt einer Zahl nur `...` als Ausgabe. Warum? Was passiert, wenn Sie den gleichen Ausdruck als Skript ausführen?

Sie müssen für diese Aufgabe kein Python-Skript abgeben, sondern lediglich die Datei `expressions.txt` mit Ihren Erklärungen.

Aufgabe 1.4 (Print-Rätsel; 3 Punkte; Dateien: `print1.py`, `print2.py` und `print3.py`)
Ihr Python-Skript soll in den folgenden Teilaufgaben aus jeweils einer Zeile der Form `print(...)` bestehen, wobei Sie `...` durch einen Ausdruck ersetzen.

(a) (1 Punkt) In der Vorlesung haben Sie *Hello-World-Programme* kennengelernt:

```
>>> print("Hello World!")
Hello World!
```

Schreiben Sie ein Python-Skript `print1.py`, das die gleiche Ausgabe wie oben produziert, dabei jedoch *kein* Leerzeichen innerhalb eines Strings enthält. Außerhalb von Strings dürfen (und sollen) Sie Leerzeichen verwenden.

Hinweis: Verwenden Sie mehrere Argumente innerhalb des `print` Befehls.

(b) (1 Punkt) Betrachten Sie folgendes Python-Programm:

```
>>> print("XXXX\nXXXX\nXXXX")
XXXX
XXXX
XXXX
```

Schreiben Sie ein Python-Skript `print2.py`, das die gleiche Ausgabe wie das obige Programm erzeugt, jedoch nur ein einziges Vorkommen des Buchstabens `X` im gesamten Python-Skript enthält.

(c) (1 Punkt) Schreiben Sie ein Python-Skript `print3.py`, das die ersten 4 der letzten 4047 Ziffern der Zahl 4047^{4047} (im Dezimalsystem) berechnet und ausgibt. Angenommen, die letzten 4047 Ziffern der Zahl wären `27180470...`, dann soll die Ausgabe wie folgt aussehen:

```
2718
```

Aufgabe 1.5 (Erfahrungen; Datei: `NOTES.md`)

Notieren Sie Ihre Erfahrungen mit diesem Übungsblatt (benötigter Zeitaufwand, Probleme, Bezug zur Vorlesung, Interessantes, etc.).

Editieren Sie hierzu die Datei `NOTES.md` im Abgabeordner dieses Übungsblattes auf unserer Webplattform. Halten Sie sich an das dort vorgegebene Format, da wir den Zeitbedarf mit einem Python-Skript automatisch statistisch auswerten. Die Zeitangabe `4.5 h` steht dabei für 4 Stunden 30 Minuten.

Der Build-Server überprüft ebenfalls, ob Sie das Format korrekt angegeben haben. Prüfen Sie, ob der Build-Server mit Ihrer Abgabe zufrieden ist, so wie es im Video zur Lehrplattform gezeigt wurde.