Einführung in die Programmierung

Prof. Dr. Peter Thiemann
Marius Weidner
Simon Dorer, Moritz Steffl

Universität Freiburg
Institut für Informatik
Wintersemester 2026

Übungsblatt 1

Abgabe: Dienstag, 27.10.2026, 10:00 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.

(a) **Setup**
Eine Anleitung zur Installation der benötigten Software finden Sie hier.

(b) **Abgabesystem**
Eine Anleitung für unser Abgabesystem finden Sie hier.

(c) **Abgaberegeln**
Die Abgaberegeln für die Übungen finden Sie hier. Beachten Sie insbesondere, dass bei einem fehlerhaften Build die gesamte Abgabe mit 0 Punkten bewertet wird.

Sollten Sie Probleme oder Fragen haben, können Sie sich gerne im Forum[1] melden oder in den Tutorien fragen.

Aufgabe 1.2 (Algorithmen; 4 Punkte; Datei `division.txt` oder `division.md`)

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 notwendigen Eigenschaften Präzision, Effektivität, statische Finitheit,

[1] Auf `https://git.proglang.de/2026WS-EidP/forum/issues` mit Ihrem RZ-Account (wie bei Ilias) einloggen.

dynamische Finitheit und Terminierung erfüllt sind. Begründen Sie jeweils kurz Ihre Antwort.

Aufgabe 1.3 (Interaktiver Modus; 3 Punkte; Datei: `repl.txt` oder `repl.md`)

Neben dem *Skript-Modus* gibt es in Python auch den *interaktiven Modus*, auch *REPL* (Read-Eval-Print Loop) genannt. In dieser Aufgabe sollen Sie die Unterschiede zwischen beiden Modi untersuchen und erklären.

Im Skript-Modus können Sie ein Python-Skript (Dateien mit der Endung `.py`) ausführen, indem Sie es in VSCode 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 im Terminal eingeben und erhalten sofort eine Ausgabe. Um den interaktiven Modus zu starten, öffnen Sie ein Terminal in VSCode und geben Sie den Befehl `python3.14` ein. Nun können Sie Python-Code hinter dem Prompt-Zeichen `>>>` eingeben:

```
>>> "Hallo Welt!"
'Hallo Welt!'
>>> print("Hallo Welt!")
Hallo Welt!
>>> (1337 + 42) // 3
459
```

Sie können den interaktiven Modus mit dem Befehl `exit()` oder der Tastenkombination `Strg` + `D` beenden.

(a) (1.5 Punkte) Wenn Sie nacheinander die Ausdrücke `42` und `print(42)` in den interaktiven Modus 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) (1.5 Punkte) Wenn Sie `print(42 + ((1337 // 2) * 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 `repl.txt` oder `repl.md` mit Ihren Erklärungen.

Aufgabe 1.4 (Print-Rätsel; 3 Punkte; Dateien: `print1.py`, `print2.py`, `print3.py`)

Ihr Python-Skript soll in den folgenden Teilaufgaben aus jeweils *einer Zeile* der Form `print(...)` bestehen, wobei Sie `...` durch einen oder mehrere Ausdrücke ersetzen.

(a) (1 Punkt) In der Vorlesung haben Sie erste Programme kennengelernt, die einen Text ausgeben:

```
>>> 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.

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

```
>>> print("AABB\nAABB\nAABB\nAABB\n")
AABB
AABB
AABB
AABB
```

Schreiben Sie ein Python-Skript `print2.py`, das die gleiche Ausgabe wie das obige Programm erzeugt. Die Buchstaben `A` und `B` sowie die Ziffern `0` bis `9` dürfen im gesamten Python-Skript höchstens ein Mal vorkommen.

(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
```

Hinweis: Überlegen Sie sich, wie Sie mit den in der Vorlesung vorgestellten Rechenoperationen `//`, `%` und `**`, diese Ziffern berechnen können.

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 Skript zum Abgabesystem gezeigt wurde.