Informatik I: Einführung in die Programmierung

Prof. Dr. Peter Thiemann
Hannes Saffrich, Simon Ging
Wintersemester 2021

Universität Freiburg
Institut für Informatik

Übungsblatt 2

Abgabe: Montag, 1.11.2020, 9:00 Uhr morgens

Hinweis zu Visual Studio Code: Auf dem ersten Übungsblatt hat bei der Konfigurationsanleitung für Visual Studio Code eine Einstellung gefehlt, die Sie noch nachträglich setzen müssen: in den Settings müssen Sie unter:
"Python > Linting: Pycodestyle Args" auf "Add Item" klicken und dort:
`--ignore=W292,E501,E704,W503,W504` eingeben. Weitere Informationen hierzu finden Sie im Forum.

Bitte beachten Sie die folgenden Hinweise, um Punktabzug zu vermeiden:

- Verwenden Sie nur Befehle und Programmiertechniken, die Inhalt der bisherigen Vorlesungen (bis zum Abgabetermin) und Übungsblättern waren.
- Nach dem Hochladen ihrer Lösung führt der Buildserver style checks ihres Codes mit `pycodestyle` aus. Diese style checks müssen mit *success* beendet werden, um volle Punktzahl zu erreichen.
- Abgaben per Mail können nicht berücksichtigt werden.
- Achten Sie auf richtige Dateiendungen: `.txt` oder `.md` für Textdateien und `.py` für Pythondateien.
- Die Textdateien sollen in *plain text* vorliegen, daher verwenden Sie am besten Visual Studio Code zur Erstellung.

An dieser Stelle möchten wir nochmal auf das Tutorial-Video zur Webplatform, Übungsabgabe und Git hinweisen, in dem die oben genannten Punkte detailliert erklärt werden.

Aufgabe 2.1 (Arithmetische Ausdrücke; Datei: `arithmetik.txt`; Punkte: 4)

Bestimmen Sie nach jeder der folgenden Wertzuweisungen an die Variable `res` den Typ von `res`. Geben Sie jeweils eine kurze Erläuterung, warum das so ist.

(a)
```
>>> from math import log2
>>> res = int(log2(64)) + 2 ** abs(1+1j)
```

(b)
```
>>> from math import sqrt, floor, ceil
>>> res = floor(2.3 * 7) * ceil(2 ** 3 + 7.1)
```

(c)
```
>>> from math import pi, sin, cos, radians
>>> res = cos(pi/4)**2 + sin(radians(45))**2j
```

(d)
```
>>> res = 6 * round(2.1, 1) // 1
```

Hinweis zu Aufgabe 2.2 und 2.3 (Text-Eingabe mit `input`**)**

In der Vorlesung wurde die Funktion `print` vorgestellt, die es einem Python-Script ermöglicht dem Benutzer eine Text-Ausgabe zu präsentieren.

Die Funktion `input` stellt das Gegenstück zu `print` dar und ermöglicht es den Benutzer nach einer Text-Eingabe zu fragen:

```
>>> s = input("Geben Sie etwas ein: ")
Geben Sie etwas ein: foo123
>>> s
'foo123'
```

Der Aufruf von `input` erzeugt dabei zunächst die Ausgabe

```
Geben Sie etwas ein:
```

und wartet dann bis der Benutzer eine beliebige Tasteneingabe tätigt (hier `foo123`) und mit einem Zeilenumbruch (Enter) die Eingabe beendet. Die vom Benutzer eingegebenen Zeichen werden dann als String zurückgegeben (hier `'foo123'`).

Aufgabe 2.2 (Celsius nach Fahrenheit; Datei: `fahrenheit.py`; Punkte: 4)

Schreiben Sie ein Python-Script `fahrenheit.py`, welches den Benutzer dazu auffordert einen Celsius-Wert (Fließkommazahl) einzugeben und anschließend den entsprechenden, auf zwei Nachkommastellen gerundeten, Fahrenheit-Wert ausgibt.

Der Aufruf des Scripts, bei dem der Benutzer den Celsius-Wert `-12.715` eingibt, soll dabei folgende Ausgabe erzeugen:

```
Celsius: -12.715
Fahrenheit: 9.11
```

Hinweis. Zum Runden können Sie die Funktion `round` verwenden. Konsultieren Sie hierzu `help(round)` im Python Interpreter.

Hinweis. Strings können wie folgt zu Fließkommazahlen konvertiert werden:

```
>>> float('2.3450')
2.345
```

Versucht man einen String, der keiner Fließkommazahl entspricht, zu konvertieren, wird die Ausführung des Programms durch eine Ausnahme zum Absturz gebracht:

```
>>> float('not a number')
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ValueError: could not convert string to float: 'not a number'
```

In Ihrem Python-Script dürfen Sie dies ignorieren. Sie können also annehmen, dass der Benutzer stehts eine Fließkommazahl eingibt.

Aufgabe 2.3 (Mantelfläche des Kegels; Datei: `kegel.py`; Punkte: 5)

Schreiben Sie ein Python-Script `kegel.py`, welches den Benutzer dazu auffordert die Höhe und den Radius eines Kegels als Gleitkommazahlen einzugeben und anschließend die Mantelfläche des Kegels, auf zwei Nachkommastellen gerundet, ausgibt.

Der Aufruf des Scripts, bei dem der Benutzer den Radius `3.0` und die Höhe `5.0` eingibt, soll dabei folgende Ausgabe erzeugen:

```
Radius: 3.0
Höhe: 5.0
Mantelfläche: 54.96
```

Aufgabe 2.4 (Abrunden; Datei: `abrunden.py`; Punkte: 5)

Schreiben Sie ein Python-Script `abrunden.py`, welches den Benutzer dazu auffordert eine beliebige Fließkommazahl einzugeben. Anschließend soll die Zahl quadriert werden und das Ergebnis ausgegeben werden.

Die quadrierte Zahl soll nun abgerundet werden. Implementieren sie hierfür mindestens 3 verschiedene Möglichkeiten. Es soll also mindestens dreimal hintereinander die Zahl abgerundet und das Ergebnis ausgegeben werden, jeweils mit einer anderen Berechnung. Sie dürfen hierfür nur Methoden aus den bisherigen Vorlesungen 1-2 sowie den Übungsblättern 1-2 benutzen.

Der Aufruf des Scripts bei dem der Benutzer die Zahl `4.5` eingibt, soll dabei folgende Ausgabe erzeugen:

```
Kommazahl: 4.5
Quadriert: 20.25
Methode 1: 20.0
Methode 2: 20.0
Methode 3: 20.0
```

Aufgabe 2.5 (Erfahrungen; 2 Punkte; 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 Webplatform. Halten Sie sich an das dort vorgegebene Format, da wir den Zeitbedarf mit einem Python-Skript automatisch statistisch auswerten. Die Zeitangabe `3.5 h` steht dabei für 3 Stunden 30 Minuten.