Übungssammlung für “Einführung in die Programmierung mit Python“

Florian Steingruber

17. April 2023

Inhaltsverzeichnis

1 Vorwort

Die vorliegende Übungssammlung ist ein Teil des Kurses „Einführung in die Programmierung mit Python“. Sie soll zur Festigung und Anwendung des theoretischen Wissens dienen. Dazu werden die vorgestellten Themen auf praktische Beispiele angewendet. Die Teilnehmenden haben dabei die Möglichkeit ihr Verständnis für den Stoff zu vertiefen. Bitte berücksichtigen Sie folgende Grundsätze:

- Grundsätzlich sind immer alle ausgearbeiteten Übungen aus vorangegangenen Kursterminen bereitzustellen, um auf bereits erarbeitete Codeteile gegebenenfalls zurückgreifen zu können.
- Sie sind bei jedem Kurstermin für die Sicherung Ihrer Daten verantwortlich.

Grundsätzlich sollten die Übungen in Einzelarbeit implementiert werden, damit jede:r Teilnehmende die gleichen Möglichkeiten hat den Umgang mit Python-Code und die Handhabung der Entwicklungsumgebung kennen zu lernen. Das Sammeln von Ideen, das Überlegen von Konzepten und Diskutieren von Implementierungsansätzen kann gerne in kleinen Gruppen erfolgen. In vielen Fällen ist der Austausch untereinander sehr hilfreich. Bei Fragen steht der Trainer auch gerne zur Verfügung. Bei Fragen zu den Unterlagen und Übungen außerhalb der Kurszeiten können Sie sich per Mail an den Trainer wenden.

Mail-Adresse: f.steingruber@gmx.at

2 Hilfreiche Hinweise für die Ausarbeitung

Diese Kapitel sammelt hilfreiche Hinweise, die bei der Ausarbeitung der Übungsbeispiele unterstützen können:

1. Verwenden Sie zur Ideensammlung Stift und Papier.
2. Schreiben Sie eine Art Lösungsidee, die die Schritte enthält, um die Aufgabe zu lösen.
3. Überlegen Sie sich welche Daten Sie verarbeiten müssen. Werden zusätzliche Daten zur Steuerung des Programmablaufs benötigt?
4. Entscheiden sie welche Datentypen geeignet sind um die Daten zu speichern. Ist eine Kombination aus mehreren Datentypen notwendig?
5. Zur Entwicklung des Programmablaufs kann das EVA-Prinzip angewendet werden.
   - **E**ingabe: Woher kommen die zuverarbeiteten Daten her? Werde diese vom Benutzer eingelesen? Stammen die Daten aus einer Datei? Welches Format haben die Daten, die aus der Datei gelesen werden? Müssen die Daten aus einer Datenbank gelesen werden?
   - **V**erarbeitung: Wie werden die Eingabedaten in die Ausgabedaten überführt? Welche Berechnungen müssen angewendet werden? Müssen die Daten umgeformt werden? Welche Algorithmen müssen angewendet werden? Soll ein eigener Algorithmus implementiert werden?
   - **A**usgabe: Wie werden die Ergebnisse zur Verfügung gestellt? Werden diese in der Konsole ausgegeben? Sollen sie als Datei zur Verfügung gestellt werden? In welchem Format werden die Daten in die Datei gespeichert?

3 Übungsbeispiele

In diesem Abschnitt können Sie die Übungsbeispiele finden.

3.1 Berechnungen am Rechteck

Mit Hilfe eines Python-Skriptes sollen der Umfang und die Fläche eines Rechtecks berechnet werden. Die benötigte Länge und Breite zur Berechnung sollen vom Benutzer abgefragt werden. Die Ergebnisse sollen in der Konsole ausgegeben werden.

3.2 Netto-Brutto-Rechner

Schreiben Sie ein Python-Skript, das es erlaubt einen gegebenen Nettobetrag in einen Bruttobetrag umzurechnen. Der Nettobetrag und der Steuersatz sollen vom Benutzer abgefragt werden. Das Ergebnis soll am Ende in der Konsole ausgegeben werden.

3.3 Kreuzprodukt für zwei 3D-Vektoren

Mit Hilfe von Python soll das Kreuzprodukt für zwei Vektoren im dreidimensionalen Raum berechnet werden. Das Kreuzprodukt für zwei Vektoren im dreidimensionalen Raum ist wie folgt definiert.

$$\vec{a} \times \vec{b} = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} a_2 \cdot b_3 - a_3 \cdot b_2 \\ a_3 \cdot b_1 - a_1 \cdot b_3 \\ a_1 \cdot b_2 - a_2 \cdot b_1 \end{pmatrix}$$

Für das Implementieren der Berechnung können die beiden Vektoren als fix angenommen werden.

3.4 Inhalt von Variablen vertauschen

Schreiben Sie ein Python-Skript, dass zwei Variablen $a = 10$ und $b = 20$ definiert. Im nächsten Schritt soll der Inhalt der beiden Variablen vertauscht werden. D.h. a bekommt den Wert von b und b den Wert von a. Geben Sie die Variablen a und b vor und nach dem Vertauschen in der Konsole aus.

3.5 Kilometer in Meilen umrechnen

Mit Hilfe von Python soll ein Skript implementiert werden, dass eine gegebenen Strecke in Kilometer in Meilen, Seemeilen und Schwedische Meilen umrechnet. Die benötigte Strecke zur Berechnung soll vom Benutzer abgefragt werden. Die Ergebnisse sollen in der Konsole ausgegeben werden.

3.6 Umkreisradius berechnen

Für ein Quadrat mit der Seitenlänge a soll der Radius für den umschreibenden Kreises berechnet werden. Die Seitenlänge a soll vom Benutzer eingegeben werden. Das Ergebnis soll in der Konsole ausgegeben werden.

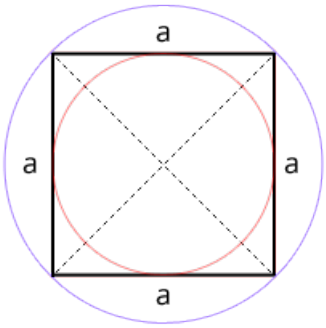

Abbildung 1: *Quadrat mit Umkreis und Inkreis*

3.7 Schaltjahr bestimmen

Schreiben Sie ein Python-Skript, das bestimmt, ob es sich bei einer gegebenen Jahreszahl um ein Schaltjahr handelt. Die Jahreszahl soll vom Benutzer abgefragt werden.

Für Fortgeschrittene: Alternativ kann die Jahreszahl als Kommandozeilenparameter an das Python-Skript übergeben werden. Dazu kann folgender Python-Code verwendet werden:

```python
# at the top of your file
import argparse

# later in your code
parser = argparse.ArgumentParser("Checks if a year is a leap year")
parser.add_argument("year", type=int, help="Year to check")

arguments = parser.parse_args()

print(f"The year {arguments.year} will be checked")
```

3.8 Heron-Verfahren zur Berechnung der Quadratwurzel

Die Quadratwurzel einer Zahl kann mit dem Heron-Verfahren angenähert werden [1]. Beim Heron-Verfahren handelt es sich um ein iteratives Verfahren und ist wie folgt definiert.

$$x_{n+1} = \frac{1}{2} \cdot \left(x_n + \frac{a}{x_n} \right)$$

Bei a handelt es sich um den Eingabewert für den die Quadratwurzel berechnet werden soll. Der Wert x_n ist der aktuelle Iterationswert. Dieser kann für die erste Iteration beliebig gewählt werden, muss aber ungleich 0 sein. Der Wert x_{n+1} ist der neu berechnete Iterationswert und stellt das Ergebnis der Quadratwurzelberechnung dar. Für jede neue Iteration wird der Wert von x_{n+1} aus dem vorhergehenden Schritt als x_n in der nächsten Iteration verwendet.

Ziel dieser Aufgabe ist es das Heron-Verfahren in Python zu implementieren. Dabei soll das Ergebnis von jedem Iterationsschritt ausgegeben werden. Legen sie für die maximale Anzahl der Iterationen einen geeigneten Wert fest. Testen Sie ihre Implementierung mit verschiedenen Werten für a. Lesen Sie den Wert über die Konsole vom Benutzer ein.

Für Fortgeschrittene: Während Sie Ihre Implementierung getestet haben, ist Ihnen vielleicht aufgefallen, dass sich das Ergebnis oft nach wenigen Iterationen nicht mehr verändert.

Daher würde es sich anbieten, anstatt der fixen Anzahl an Iterationen ein anderes Abbruchkriterium für die Berechnung zu verwenden. Überlegen Sie sich, welches Abbruchkriterium Sie verwenden könnten. Gibt es hier etwas, das berücksichtigt werden muss? Verwenden Sie in Ihrer bestehenden Implementierung nun ihr definiertes Abbruchkriterium. Zählen Sie die Anzahl der benötigten Iterationen und geben Sie diese in der Konsole aus.

3.9 Pluperfect Digital Invariant

Bei einer PPDI (Pluperfect Digital Invariant) handelt es sich um eine narzisstische Zahl, deren Summe ihrer Ziffern, jeweils potenziert mit der Stellenanzahl der Zahl, wieder die Zahl selbst ergibt [2].

Schreiben Sie ein Python-Skript, das eine gegebene Zahl a überprüft, ob es sich bei dieser Zahl a um eine PPDI handelt. Folgende Bedingung muss dazu erfüllt sein.

$$a_n^n + a_{n-1}^n + a_{n-2}^n + \cdots + a_1^n = a$$

Die zu überprüfende Zahl kann vom Benutzer eingelesen werden oder alternativ als Kommandozeilenargument an das Python-Skript übergeben werden. Entscheiden Sie selbst, welche Variante Ihnen besser gefällt.

3.10 Ratespiel

Implementieren Sie ein Ratespiel in Python. In diesem Ratespiel soll der Spieler eine geheime Zahl zwischen 1 und 100 erraten. Bei der geheimen Zahl handelt es sich um eine zufallsgenerierte Zahl, die zu Spielbeginn ermittelt wird. Während des Spiels wird der Spieler aufgefordert einen Tipp abzugeben. Überprüfen Sie hier, ob sich die Eingabe des Benutzers im gültigen Zahlenbereich befindet. Liegt der Tipp des Benutzers daneben, bekommt der Benutzer einen Hinweis, ob die Zahl größer oder kleiner als die geheime Zahl ist. Das Spiel ist dann zu Ende, wenn die Zahl erraten wurde. Um eine Zufallszahl zu generieren, kann folgender Code-Ausschnitt verwendet werden.

```
1 # at the top of your file
2 import random
3
4 # later in your code
5 secret_number = random.randint(1, 100)
```

3.11 Zahlen verdrehen

Schreiben Sie ein Python-Skript, das die Zahlen von 1 bis inkl. 30 in umgekehrter Reihenfolge in der Konsole ausgibt.

3.12 Kassenbericht für ein digitales Kassensystem

Ein digitales Kassensystem speichert alle Verkäufe eines Tages in einer csv-Datei. In der csv-Datei werden die Verkäufe sequentiell nach Bezeichnung, Warengruppe und Preis aufgelistet. Ein Ausschnitt aus einer solchen csv.Datei ist im Abschnitt unterhalb zu sehen.

```
Bananen;A;1.50
Mineralwasser;B;3.00
Backpapier;C;1.00
Mehl;A;0.50
Bleistifte;C;2.50
Mehl;A;0.50
Backpapier;C;1.00
...
```

Leider fehlt dem Kassensystem noch die Funktionalität einen Kassenbericht automatisiert zu erstellen. Ihre Aufgabe ist es, für dieses Kassensystem eine Software in Python zu schreiben, die auf Basis der bestehenden csv-Datei einen solchen Kassenbericht erstellt. Die folgenden Informationen sollen im Kassenbericht enthalten sein:

- Menge aller verkauften Artikel
- Summe aller Verkäufe
- Summe der Verkäufe nach Warengruppe

Damit der Kassenmitarbeiter den Kassenbericht korrekt in das Kassenbuch eintragen kann, soll der Kassenbericht visuell dargestellt werden. Für die erste Implementierung reicht eine rein textuelle Ausgabe (in der Konsole).

Für Fortgeschrittene: Für die digital Weiterverarbeitung soll die csv-Datei in eine komprimierte Form überführt werden. D.h. gleiche Waren werden zusammengefasst und die Menge und der Gesamtpreis angegeben. Die Verkäufe werden in der komprimierten Form nach Menge, Bezeichnung, Warengruppe und Gesamtpreis aufgelistet. Die generierte Datei könnte wie folgt aussehen:

```
1;Bananen;A;1.50
1;Mineralwasser;B;3.00
2;Backpapier;C;2.00
2;Mehl;A;1.0
1;Bleistifte;C;2.50
...
```

3.13 Erweiterung für das Modul statistics

Während des Vortrags haben wir ein Python-Modul *statistics* begonnen zu schreiben. Dieses Modul enthält bereits die Funktionen *mean(...)* und *mode(...)*. In dieser Übung soll das Modul um eine Funktion *median(...)* erweitert werden, die den Median für eine übergebene Liste an Zahlen bestimmt. **Hinweis:** Achten Sie darauf, dass die übergebene Liste durch den Aufruf der Funktion nicht verändert werden darf.

3.14 Temperaturumrechnung

Schreiben Sie ein Python-Modul, das Funktionen zur Umrechnung zwischen verschiedenen Temperaturskalen anbietet. Folgende Umrechnungen sollen möglich sein:

- Celsius in Fahrenheit

- Fahrenheit in Celsius
- Celsius in Kelvin
- Kelvin in Celsius

Achten Sie bei der Implementierung auf die physikalischen Grenzen.

3.15 Textanalyse

Ziel dieses Beispiels ist es, eine einfache Textanalyse durchzuführen. Dazu soll eine Funktion *analyze_text* geschrieben werden, die einen Text wortweise analysiert. Der Text soll aus einer Textdatei gelesen werden. Der Dateiname wird der Funktion als Parameter übergeben. Für den eingelesenen Text sollen die folgenden Metriken bestimmt werden:

- Die Anzahl der Wörter im Text
- Die durchschnittliche Wortlänge
- Das kürzeste Wort und dessen Länge
- Das längste Wort und dessen Länge

Die Analyseergebnisse sollen in der Konsole ausgegeben werden.

3.16 Frequenzanalyse von Texten

Ein Verfahren in der Kryptoanalyse ist die sogenannte Frequenzanalyse. Dabei wird z.B. für einem Text die Häufigkeit der einzelnen Buchstaben, die im Text vorkommen, berechnet. Mit Hilfe der Frequenzanalyse kann zum Beispiel die Verschiebungschiffre gebrochen werden. Schreiben Sie nun ein Python-Skript, dass für eine Textdatei eine Frequenzanalyse durchführt. Die Texte sind dabei auf die englische Sprache begrenzt. Geben sie die berechneten Häufigkeiten der einzelnen Buchstaben in der Konsole aus.

3.17 Palindrome erkennen

Schreiben Sie ein Python-Skript, das erkennt, ob es sich bei einem Wort um ein Palindrom handelt. Das zu untersuchende Wort soll vom Benutzer eingelesen und das Ergebnis in der Konsole ausgegeben werden. Unter einem Palindrom versteht man Wörter oder Sätze, die rückwärts gelesen den gleichen oder einen sinnhaften Text ergeben. Wir beschränken uns in diesem Beispiel auf Wörter die vorwärts und rückwärts gelesen das gleiche Wort ergeben. Mögliche Beispiel sind:

- Anna
- Otto
- Ebbe
- Rentner
- Reittier
- Lagerregal
- Retsinakanister

3.18 Verschlüsselung mit Morse-Code

Für die Verschlüsselung von Texten kann neben anderen Verschlüsselungsverfahren, wie z.B. RSA, auch der Morse-Code verwendet werden. Schreiben Sie ein Python-Skript, das es erlaubt, eine gegebene Textdatei mit Hilfe des Morse-Codes zu verschlüsseln. Um die verschlüsselte Textdatei wieder in ein lesbares Format zu bringen, soll auch die Entschlüsselung implementiert werden. Um die Robustheit und Korrektheit Ihrer Implementierung zu testen, können Sie eine Ihrer verschlüsselten Dateien mit anderen Teilnehmer:innen austauschen und versuchen diese mit Ihrer Implementierung zu entschlüsseln. Für die Verschlüsselung und Entschlüsselung kann die nachstehende Tabelle 1 verwendet werden. Um die einzelnen kodierten Zeichen voneinander trennen zu können, können sie zwischen zwei kodierten Zeichen ein Leerzeichen platzieren. Zwischen zwei kodierten Wörtern können Sie drei Leerzeichen einfügen.

A	.-	B	-...	C	-.-.	D	-..	E	.	F	..-.	G	–.
H	....	I	..	J	.—	K	-.-	J	.-..	M	–	N	-.
O	—	P	.–.	Q	–.-	R	.-.	S	...	T	-	U	..-
V	...-	W	.–	X	-..-	Y	-.–	Z	–..	Ä	.-.-	Ö	—.
ß	.....	Ü	..–										
0	——	1	.—-	2	..—	3	...–	4	....-	5	.....	6	-...
7	–...	8	—..	9	—-.								
.	.-.-.-	,	–..–	?	..–..	’	.—-.	!	-.-.–	/	-..-.	(	-.–.
)	-.–.-	&	.-...	:	—...	;	-.-.-.	=	-...-	+	.-.-.	-	-....-
_	..–.-	”	.-..-.	$	...-..-	@	.–.-.						

Tabelle 1: Morsealphabet

3.19 Wetterstation

Schreiben Sie eine Klasse zur Darstellung der Daten einer Wetterstation. Die Klasse soll folgende Daten speichern können:

- Name der Wetterstation
- Temperatur in °C
- Luftfeuchtigkeit in %

Die gespeicherten Daten sollen über den Konstruktor initialisiert werden. Für den Zugriff auf die Daten sollen “Getter“- und “Setter“-Funktionen oder Properties verwendet werden. Die Entscheidung liegt bei Ihnen. Die Temperatur soll sowohl in °C als auch in °F gesetzt und gelesen werden können. Gespeichert wird sie, wie oben bereits steht, nur in °C. Implementieren Sie zusätzlich eine Funktion *(*pring_data), die die Daten der Wetterstation in der Konsole ausgibt. Die Ausgabe enthält den Namen, die Temperatur in °C, die Temperatur in °F und die Luftfeuchtigkeit. Schreiben sie ein kurzes Python-Skript, um Ihre Implementierung zu testen. Achten sie auf die Wertebereiche der gespeicherten Daten!

3.20 Verwaltung von Wetterstationen

Im vorherigen Beispiel wurde eine Klasse zur Darstellung von Wetterstationen implementiert. Schreiben Sie nun eine Klasse, die es erlaubt mehrere Wetterstationen zu verwalten. Es soll

möglich sein Wetterstationen hinzuzufügen, jedoch nur, wenn noch keine Wetterstation mit dem gleichen Namen enthalten ist. Wetterstationen mit einem bestimmten Namen sollen gelöscht werden können. Die Anzahl an Wetterstationen, die gespeichert werden können, soll begrenzt sein. Dieser Wert kann über den Konstruktor mitgegeben werden.

Die folgenden Abfragen sollen mindestens mit Ihrer implementierten Klasse möglich sein:

- Anzahl der hinzugefügten Wetterstationen
- Ausgabe aller hinzugefügten Wetterstationen
- Wetterstation mit der niedrigsten Temperatur
- Wetterstation mit der höchsten Temperatur

Für Fortgeschrittene: Die Wetterstationen sollen aufsteigend nach ihrem Namen sortiert gespeichert werden.

3.21 Mitarbeiterverwaltung

Ein Unternehmen benötigt eine Software für die Verwaltung ihrer Mitarbeiter. Es gibt verschiedene Arten von Mitarbeitern. Für jede Art von Mitarbeiter wird das Gehalt unterschiedlich berechnet.

Jeder Mitarbeiter hat: einen Vor- und einem Nachnamen, ein Namenskürzel (3 Buchstaben), eine Sozialversicherungsnummer, ein Geburtsdatum und ein Eintrittsjahr (das Jahr in dem der Mitarbeiter begonnen hat im Unternehmen zu arbeiten).

Bei der Gehaltsberechnung wird unterschieden zwischen:

- *CommissionWorker*: Grundgehalt + Fixbetrag pro verkauftem Stück
- *HourlyWorker*: Stundenlohn x gearbeitete Stunden pro Monat
- *PieceWorker*: Summe erzeugter Stücke x Stückwert
- *Boss*: monatliches Fixgehalt

Überlegen Sie sich, welche Member und Funktionen die einzelnen Klassen benötigen, um mindestens folgende Abfragen zu ermöglichen:

- Wie viele Mitarbeiter hat das Unternehmen?
- Wie viele *CommissionWorker* arbeiten in der Firma?
- Wie viele Stück wurden im Monat erzeugt?
- Wie viele Stück wurden im Monat verkauft?
- Wie viele Mitarbeiter vor 1970 geboren?
- Gibt es einen Mitarbeiter zu einem gegebenen Namenskürzel?

- Welche(r) Mitarbeiter ist/sind am längsten im Unternehmen?
- Ausgabe aller Datenblätter der Mitarbeiter

Zur Vereinfachung braucht nur ein Monat berücksichtigt werden (d.h. pro Mitarbeiter nur ein Wert für Stückzahl oder verkaufte Stück). Die Ausgabe hat folgendes Aussehen:

```
****************************************
Fa. Hofer, Linz
****************************************
Datenblatt
---------------
Name: Max Huber
Kürzel: mhu
Sozialversicherungsnummer: 1234010273
Einstiegsjahr: 2005
Mitarbeiterklasse: CommissionWorker
Grundgehalt: 2500 EUR
Provision: 350 EUR
Gesamtgehalt: 2850 EUR
----------------------------------------
v1.0 Oktober 2022
----------------------------------------
```

Testen Sie Ihre Implementierung mit einer Klasse *Client*, die die Abfragen ausführt und die Ergebnisse ggf. ausgibt.

Für Fortgeschrittene: Es wird eine neue Art von Mitarbeiter *Coach* eingeführt. Die Aufgabe eines *Coach* ist das Führen von Mitarbeitern. Daher hat jeder Coach eine Liste an Mitarbeitern, für die er/sie verantwortlich ist. Diese Liste soll im Datenblatt ausgegeben werden. Der Name der jeweiligen Mitarbeiter ist ausreichend. Das Gehalt eines *Coach* berechnet sich wie folgt: monatliches Fixgehalt + Anzahl der geführten Mitarbeiter x 50 €.

Literatur

[1] wikipedia.org - Heron-Verfahren, `https://de.wikipedia.org/wiki/Heron-Verfahren`, 02. November 2022 21:41 Uhr

[2] wikipedia.com - Narzisstische Zahl, `https://de.wikipedia.org/wiki/Narzisstische_Zahl`, 02. November 2022 17:14 Uhr