Einführung in die Programmierung

Prof. Dr. Gemma Roig
Prof. Dr. Matthias Kaschube
Dr. Karsten Tolle

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Übungsblatt ÜE-09

Ausgabe: 28.01.2026
Abgabe: 07.02.2026
12:00 Uhr

mit *Code Review* - einzeln!

Hinweise:

- *Sie dürfen für dieses Blatt alles nutzen, jedoch ist es nicht zielführend automatisch generierte Lösungen (z.B. über ein LLM) einfach zu kopieren und hochzuladen.* ***Quellen sind anzugeben, also auch ob ein LLM bei der Erstellung unterstützt hat!***
- *Dieses Blatt ist einzeln zu lösen! Mit der __author__ Angabe (siehe nächster Punkt) bestätigen Sie, dass die Abgabe Ihr Werk ist. Die Abgaben werden auch auf Plagiate geprüft. Sollten wir identische Abgaben finden, erhält keiner der abgebenden Punkte. Sollten Sie die Arbeit mit anderen Personen zusammen erstellen oder auch mit der Unterstützung von LLMs, sollte dies in der __credits__ Variable vermerkt sein. In diesem Fall würde identischer Code noch OK sein. Wichtig ist aber, dass jeder (auch bei Zusammenarbeit) seine eigenen Kommentare erstellt. Sind diese also auch gleich, wird es trotz Angabe von möglichen __credits__ als Plagiat (0 Punkte) gewertet.*
- *Geben Sie am Anfang aller Ihrer Quellcode Dateien (.py) die Variable* ***__author__*** *an (am Anfang des Quellcodes) – ist schon mal eine Übung, da dies für die Code Review-Aufgaben, welche für die Studienleistung relevant sind wichtig sein wird:*
  __author__ = "<Matr-Nr>, <Nachname>" also z.B.:
  __author__ = "1234567, Tolle"
- *Digitale Abgaben, die nicht im Format* ***.pdf*** *oder* ***.txt*** *für Texte oder* ***.py*** *für Code erfolgen, werden nicht überprüft. Bei Abgaben mehrerer Dateien müssen diese als* ***.zip*** *zusammengefasst werden.*
- *Sollten Aufgabenstellungen nicht eindeutig sein, bitte durch Ihre eigenen Annahmen ergänzen und diese dokumentieren (z.B. in einem Kommentar).*
- *Achten Sie auch auf Vorgaben Ihrer Tutorin/Tutors. Hierfür auch in das Gruppenforum im Moodle-Kurs von EPR schauen!*
- *Beim Programmieren und Kommentieren halten Sie sich bitte an die Regeln im Programmierhandbuch (Style Guide), siehe Moodle-Kurs. Im Zweifelsfall gilt PEP 8. … möglichst selbst mit pycodestyle mal prüfen. Nicht einhalten der Konventionen führt auch zu Punktabzug.*
- *Doc-Strings sind für jedes Modul, Funktion, Klasse, Methode, … zwingend anzugeben.*
- *Achten Sie darauf, dass Ihr Code* ***lauffähig*** *ist. Bereiche/Funktionalitäten, die nicht lauffähig sind, werden standardmäßig mit 0 Punkten bewertet.*
- *Datei- und Ordnernamen sollen keine Umlaute, diakritische oder Sonderzeichen, mit Ausnahme des Unterstrichs, enthalten!*

- *Ihre Lösung muss ohne Installation von nicht in Python 3.1x enthaltenen Standard-Bibliotheken auskommen. Die Korrektur und Ausführung durch die Tutoren muss also* **ohne** *ein „pip install …" auskommen.*

Für dieses Blatt können bis zu 20 Punkte erreicht werden. Neben der Erfüllung der Aufgabenstellung durch entsprechender Abgabe im Moodle-Kurs, ist ein Code Review bei Ihrer Tutorin/Tutor notwendig. → kein code review = 0 Punkte
Im Code Review sollen Sie Ihre Lösung vorstellen. Dies sollte vorbereitet werden und ist Teil der Aufgabe, sprich wird ebenfalls mit bewertet. Bereiche, die im Codereview nicht erklärt werden können, werden ebenfalls mit 0 Punkten bewertet.

1. Aufgabe – Tkinter: PW-Generator

Erstellen Sie in Python 3.1x und Tkinter eine Graphische Benutzungsoberfläche (GUI), mit welcher Passwörter erstellt werden können. Eine Beispielfunktion zum Erstellen von Passwörtern an sich wird zur Verfügung gestellt (Datei: generator.py). Wer möchte, kann auch eine eigene Funktion nutzen, jedoch müssen mindestens die folgenden Anforderungen erfüllt sein und über die GUI „gesteuert" werden können:

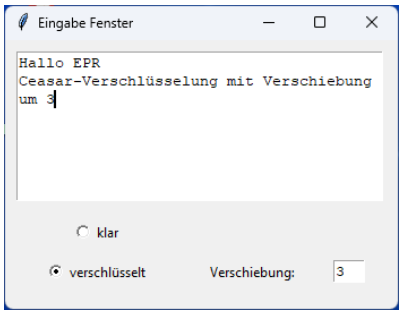
a. Länge des Passwortes,

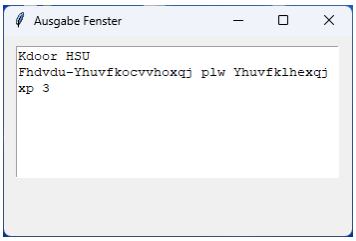
b. Angabe, welche Zeichensätze (lower, upper, digits, …) mit genutzt werden sollen oder nicht.

Anforderung an Umsetzung ist weiterhin, dass der Code der GUI getrennt ist (ein einer eigenen .py Datei) vom Code für die Passworterstellung an sich.

2. Aufgabe – Tkinter: Verschlüsselte Fenster

Erstellen Sie in Python 3.1x und Tkinter eine Graphische Benutzungsoberfläche (GUI), welche zwei Fenster erzeugt. Beide Fenster sollen ein Textfeld beinhalten. In einem der Fenster soll es weiterhin möglich sein zu wählen, ob der Text, welcher im Textfeld eingetragen wird unverschlüsselt oder verschlüsselt auf das andere Fenster übertragen wird. Als sehr einfache Verschlüsselung kann z.B. die Ceasar-Verschlüsselung (https://de.wikipedia.org/wiki/Caesar-Verschl%C3%BCsselung) dienen. Der Fokus liegt auf der GUI, nicht auf der Verschlüsselung☺
Unten beispielhaft dargestellt, muss aber nicht genauso aussehen.





Auch hier bitte eine klare Trennung von Funktionalität (Verschlüsselung) und GUI in getrennte .py-Dateien.

Anmerkung für die Aufgaben 1) und 2): Doctests sind für diese Aufgaben nicht vorgesehen. Für das Testen von GUI-Entwicklungen sind diese auch wenig sinnvoll. Hier reicht ein manuelles Testen … also bekomme ich auch angezeigt, was ich mir vorgestellt habe. In einer **Dokumentationsdatei** sollten daher mindestens **Screenshots der erstellten GUIs** mit abgeben werden.

Aufgabe 3: Numpy und Aufgabe 4: K-Means siehe → Übungsblatt_EPR_9.ipynb