# Einführung in die Programmierung

Prof. Dr. Detlef Krömker
Alexander Wolodkin

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

## Übungsblatt Nr. 7[1]

Ausgabe: 17.01.2018
Abgabe: 03.02.2018
16:00 Uhr

# Huffman-Optimalcodierung

**Hinweis:** *Dieses Aufgabenblatt ist mit einer Bearbeitungszeit von zwei Wochen dafür ausgelegt in einem Zweierteam gelöst zu werden. Neben der Implementierung wird bei der Bearbeitung Wert auf die folgenden Punkte gelegt: Dokumentation[2], Strukturierung und Einhalten des Style-Guides[3].*

**Achtung:** Achten Sie darauf die Variable `__author__` in **allen** Quellcode Dateien **korrekt** zu setzen. Abgaben die nicht dieser Vorgabe entsprechen werden **nicht** bewertet! Quellcode muss als *.py* Datei und alles Weitere als *.pdf* Datei abgegeben werden. Bei Abgaben mehrer Dateien müssen diese als *.zip* zusammengefasst werden. Abgaben, die nicht diesen Regeln entsprechen, werden ebenfalls **nicht** bewertet! Außerdem muss ihr Name in jeder abgegebenen *.pdf* Datei zu finden sein. Jeder von Ihnen MUSS bei EPR05 oder EPR06 für den Rest des Semesters den Partner wechseln! Sollten Sie dies nicht tun, werden die Aufgaben nicht bewertet.

Σ __ / **16**

### **Aufgabe 7.1**: **Einleitung**

Schreiben Sie ein Python 3.6+ - Programm das

- verschiedene Basistexte analysiert,
- hierfür eine Huffman-Optimalcodierung ermittelt,
- diese im csv-Format abspeichert,
- einen codec (Coder-Decoder) implementiert
- und diesen an einem weiteren Text ausprobiert.

Die Entwicklung soll **streng objektorientiert** erfolgen. Beachten Sie bitte: Es gibt natürlich nicht nur eine Lösung. Für alles das, was nicht angegeben ist, haben Sie die **Freiheit und Pflicht** dies festzulegen. Dokumentieren Sie diesen Schritt.

**User-Stories** (Auszug aus dem Lastenheft):

1. Ich möchte ein oder mehrere ASCII Textfiles (+ Umlaute und Eszett) als Basistexte einlesen können.
2. Ich möchte diese Textfiles bezüglich der Zeichenhäufigkeit analysieren bekommen.
3. Das Ergebnis möchte ich als csv-Datei in sortierter Form abspeichert bekommen. Die Datei habe folgende Struktur:

[1] Es dürfen keine Lösungen aus dem Skript, dem Internet oder anderen Quellen abgeschrieben werden. Diese Quellen dürfen nur mit Quellenangaben verwendet werden und es muss ein hinreichend großer Eigenanteil in den Lösungen deutlich zu erkennen sein.

[2] `http://www.python.org/dev/peps/pep-0257`

[3] `https://www.python.org/dev/peps/pep-0008/`

| Zeichen | Absolute Häufigkeit | Relative Häufigkeit | Informationsgehalt dieses Zeichens |
|---|---|---|---|
| a | 90 | 0.061 | $< Wert >$ |
| A | 15 | 0.010 | ... |
| ... | ... | ... | ... |
| Satzzeichen etc. | ... | ... | ... |

4. Auf dieser Basis möchte ich meinen persönlichen für diese Basistexte **optimalen** Huffman Code entwickelt bekommen und diesen in einer fünften Spalte in der Tabelle nach (3) hinzugefügt bekommen. Der Code soll ein Binärcode sein und verlustfrei aber optimal kodieren.
5. Für den so entwickelten optimalen Huffman Code möchte ich einen codec (Coder und Decoder) entwickelt bekommen. Dieser soll den Code aus einer Tabelle nach (4) einlesen und benutzen. Der Codec erhält entweder eine Textdatei als Input und erzeugt daraus ein binär-kodierten Huffman File oder vice versa.
6. Der Codec soll in allen Fällen den erreichten Kompressionsgrad auf der Konsole ausgeben.

**Aufgabe 7.2**: Punkte: ____ / 5

Legen sie ein *User-Handbuch* (als Bedienhilfe) und ein *Technisches Handbuch* (in dem z.B. das konkrete Fileformat und das Klassendiagramm dokumentiert sind) an. Diese beiden Dokumente sollen Sie während der Entwicklung pflegen und mit der Software abgeben. Die zu entwickelnde Software soll ein Programm sein und unter einem User-Interface laufen. Vor der Abgabe sollten Sie im technischen Handbuch Ihren Entwicklungsstand dokumentieren und auch z.B. bekannte Bugs benennen.

Im Zuge der Software-Entwicklung muss ein UML-Klassendiagramm entstehen und im Handbuch dokumentiert werden. Hierbei sollen für alle Klassen mindestens alle public-Attribute und Methoden angegeben sein. Denken Sie auch daran die Multiplizitäten anzugeben. Ggf. auch Kommentare.

**Aufgabe 7.3**: Punkte: ____ / 2

Als Basis texte für die Erstellung des Codebuchs wählen Sie:

- Friedrich Schiller: „Das Lied von der Glocke“ und
- Johann Wolfgang Goethe: „Eine Gespenstergeschichte“.

Schreiben Sie eine Software, die diese beiden Texte einliest, bezüglich der Zeichenhäufigkeit analysiert und eine Tabelle gemäß User-Story (3) speichert: Für jeden Text separat und diese beiden als „Gesamttext“.

**Achtung**: Unterscheiden Sie den einfachen Zeilenumbruch beim Gedicht und die Markierung eines Absatzes!

Neben diesen beiden Zeichen beziehen alle sonstigen Zeichen (Satzzeichen, Apostroph, etc.) in unsere Analyse mit ein und kodieren Sie auch diese.

Gibt es Auffälligkeiten bei den Häufigkeitsverteilungen zwischen den Texten?

Sie dürfen und sollen Ihre Entwicklungen zum Filehandling aus EPR_06 durchaus nutzen.

**Aufgabe 7.4**: Punkte: ____ / 3

Schreiben Sie eine Python 3-Software, die einen **optimalen** Huffman Code für eine eingelesene csv-Datei nach Aufgabe 7.3 berechnet und ergänzt. Denken Sie daran, z.B. eine Entscheidung darüber zu treffen, was passiert, wenn eine Datei schon einen Code enthält.

**Aufgabe 7.5**: Punkte: ____ / 5

Schreiben Sie eine Python 3-Software, die den Codec implementiert und dazu entweder Texte oder Huffman codierte Binärfiles einliest und das Gegenstück dazu ausgibt.

**Aufgabe 7.6**: Punkt: ____ / 1

Wenden Sie Ihre in 7.5 entwickelte Software für folgende Fälle an und dokumentieren Sie in einem Ergebnisdokument die erreichten Kompressionsgrade für den Beispieltext von Theodor Fontane: „John Maynard“ mit den drei verschiedenen Codetabellen nach 7.4.

Alle Beispieltexte sind beigefügt.