Institut für Informatik
Dr. Tobias Neckel
Alfredo Parra
Paul Sârbu

Technische Universität München
WS 2016/2017

Einführung in die wissenschaftliche Programmierung

Übungsblatt 5

1.) Einfache Ein– und Ausgabe

Gegeben ist eine Datei `input.txt`, die zeilenweise eingelesen werden soll. Jede Zeile der Datei enthält eine Zahl als String im ASCII–Format. Ermitteln Sie das Maximum und das Minimum der in der Datei enthaltenen Zahlen und schreiben Sie diese in eine neue Datei `output.txt`. Bei der Eingabedatei `input.txt`, die Sie auf der Webseite zur Veranstaltung finden, soll die Ausgabedatei wie folgt aussehen:

```
max: 42.69
min: -1.0
```

2.) Caesar Verschlüsselung (Hands-on)

Auf Blatt 2 haben Sie die Caesar Verschlüsselung kennengelernt. Die dadurch verschlüsselten Texte können leicht entschlüsselt werden. Schreiben Sie ein Python Programm, welches einen Text aus einer Datei einliest und feststellt, welcher "shift-Wert" `s` verwendet wurde. Gehen Sie hierfür davon aus, dass `e` der häufigste Buchstabe in der verschlüsselten Sprache ist. Auf der Webseite zur Veranstaltung finden Sie drei verschlüsselte Texte (`text[1-3].txt`). Testen Sie Ihr Programm mit diesen Texten, indem Sie

i) zählen, wie oft jeder Buchstabe (a-z) im verschlüsselten Text vorkommt, und

ii) feststellen, welcher Buchstabe am häufigsten vorkommt.

Dieser Buchstabe entspricht also mit großer Wahrscheinlichkeit dem Buchstabe `e` geshiftet um `s`.

Fortgeschritten

Diese Methode (`e` häufigster Buchstabe) scheitert häufig bei kurzen Texten. Auf der Webseite zur Veranstaltung finden sie zwei weitere verschlüsselte Texte (`fort-text[1-2].txt`), die mit der oben beschriebenen Methode nicht zu entschlüsseln sind.
Um auch diese Texte zu entschlüsseln, können Sie die Datei `top10000en.txt`, welche die 10,000 häufigsten Wörter der englischen Sprache enthält, für Ihren Algorithmus verwenden. Versuchen Sie, den shift-Wert `s` mit Hilfe dieser Wortliste zu bestimmen, indem Sie alle 26 Entschlüsselungsmöglichkeiten ausprobieren. Die Variante, bei der die meisten bekannten (also in der Wortliste enthaltenen) Wörter in dem entschlüsselten Text gefunden werden entspricht dann mit sehr hoher Wahrscheinlichkeit dem korrekten gesuchten shift-Wert.
Benutzen Sie brauchbare Funktionen der Python Standard Library. Die `find` Methode von Strings hat sich als nützlich erwiesen: wenn Sie einen String `text = 'abcdef'` haben, können Sie darin nach einem Suchwort 'bc' mit `text.find('bc')` suchen. Zudem können Sie z.B. mit `a = 3` und `text.find('bc', a)` bestimmen, dass erst ab der 3. Stelle in `text` gesucht werden soll. Genaueres zu dieser Methode finden Sie in der Dokumentation der Python Standard Library unter

http://docs.python.org/library/

3.) Numerische Quadratur revisited

Auf Blatt 4 haben Sie die Rechtecks- und Trapezregel als Funktionen implementiert, welche die Summe

$$I_Q(f) = \sum_{i=1}^{n} w_i f(x_i) \tag{1}$$

berechnen. Nun sollen zwei *Klassen* `Midpoint` und `Trapezoidal` mit Hilfe der jeweiligen Funktionen konstruiert werden. Die beiden Klassen sollen folgende Methoden implementieren:

- `__init__(self,a,b,n)`:
  Konstruktor, initialisiert die Klassenattribute `a`, `b` $\in \mathbb{R}$ (Integrationsintervall), und `n` $\in \mathbb{N}$ (Anzahl von Integrationspunkten).

- `constructMethod(self)`:
  gibt zwei Listen `x` und `w` zurück, welche den Variablen x und w in Formel (1) entsprechen.

- `integrate(self,f)`:
  bekommt eine Funktion `f` als Parameter, ruft `constructMethod()` auf, berechnet die Summe (1), und gibt sie zurück.

Wie unterscheiden sich die Klassen `Midpoint` und `Trapezoidal`? Was haben sie gemeinsam? Lässt sich der Code leicht erweitern (z.B. eine neue Quadraturregel einführen)?
Testen Sie Ihre Klassen, indem Sie Objekte mit verschiedenen Parametern erzeugen, die dann die Methode `integrate` aufrufen.

Fortgeschritten: Bauen Sie die spezielle Methode `__call__()` in Ihren Code ein, um den Aufruf von `integrate` zu erleichtern.