Institut für Informatik
Scientific Computing and Bioinformatics
Prof. Dr. Andreas Hildebrandt
M.Sc. Jennifer Leclaire

JOHANNES GUTENBERG UNIVERSITÄT MAINZ

Einführung in die Programmierung - WS 2016/2017

6. Übungsblatt

Abgabe: 09.12.2016 - 10:00

Aufgabe 1: Arrays und built-in Funktionen *(2 Punkte)*

Legen Sie ein Array an und füllen Sie es mit 100 Zufallszahlen vom Typ `float`. Dabei sollen die Zufallszahlen zwischen 0 und 1000 liegen. Bestimmen Sie anschließend folgende Werte:

- das kleinste Arrayelement
- das größte Arrayelement
- die Summe aller Arrayelemente
- den Mittelwert aller Arrayelemente

Berechnen Sie die Werte einmal ohne und einmal unter der Verwendung von **built-in** Funktionen. Geben Sie anschließend jeweils beide Werte separiert durch ein Tab aus.
Hinweis: Für die Berechnung des Mittelwerts gibt es keine built-in Funktion in Python, verwenden Sie daher aus dem Modul **Statistics** die Funktion **mean()**. Das Modul gibt es nur ab der Python-Version 3.4.

Aufgabe 2: Horner-Schema für Polynome *(3 Punkte)*

Schreiben Sie ein Programm, das ein Polynom

$$p(x) = a_o + a_1 x + a_2 x^2 + \ldots + a_n x^n = \sum_{i=0}^{n} a_i x^i$$

einlesen und an einer Stelle x auswerten kann. Das Program soll zunächst den Grad des Polynoms $n \in \mathbb{N}$ und dann die Koeffizienten $a_i \in \mathbb{R}$ nacheinander $(i = 0, \ldots, n)$ von der Kommandozeile einlesen. Zuletzt soll die Stelle $x \in \mathbb{R}$, an der das Polynom ausgewertet werden soll eingelesen werden. Wenden Sie nun das Horner-Schema zur Auswertung des Polynoms p an der Stelle x an und geben Sie den Wert aus. Das Horner-Schema für Polynome beruht auf folgender Identität:

$$\begin{aligned} p(x) &= a_o + a_1 x + a_2 x^2 + \ldots + a_{n-1} x^{n-1} + a_n x^n \\ &= a_0 + x(a_1 + x(a_2 + \ldots + x(a_{n-1} + x \cdot a_n) \ldots)) \end{aligned}$$

Aufgabe 3: Wachstumsverhalten von Funktionen II *(3 Punkte)*

Schauen Sie sich noch einmal die Übungsaufgabe 3 vom 4. Übungsblatt an. Darin sollten Sie für bestimmte Werte Funktionswerte der unten stehenden Funktionen berechnen und ausgeben.
Legen Sie für jede der Funktionen ein Array an und berechnen Sie die Funktionswerte für die Werte `n = 2, 3, 4, 5, ... 12`, die in den entsprechenden Arrays zu speichern sind.
Sie können die Arrays mit der Bibliothek *matplotlib* plotten. Der Plot soll eine Legende haben und jede der Graphen ein Label, das in der Legende angezeigt wird. Ein Beispiel für eine solche Darstellung

für die Funktion $f(n) = n$ finden Sie auf der nächsten Seite. Fertigen Sie zwei Plots: Ein Plot soll eine lineare Skala für die Ordinate aufweisen, der zweite eine logarithmierte Skala. Geben Sie beide Plots im *png* Format ab.

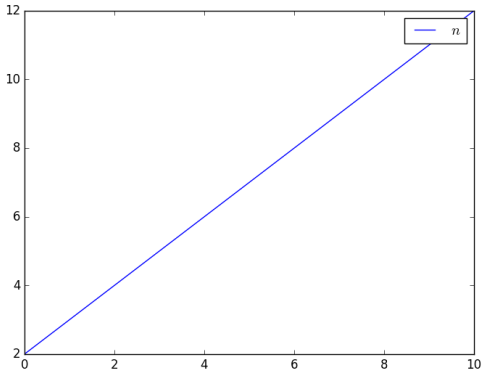
a) $log_2(n)$

b) $n \cdot log_e(n)$

c) n^2

d) n^3

e) 2^n



Hinweis: Anders als in der Vorlesung müssen Sie am Ende des Programms die Funktion **matplotlib.pyplot.show()** aufrufen, um den Plot angezeigt zu bekommen.
Hinweis: Um die Bibliothek *matplotlib* verwenden zu können, müssen Sie sie zunächst installieren und einbinden. Eine Anleitung dazu finden Sie in der Aufgabe 5 *conda und pycharm* auf dem 0. Übungsblatt (Präsenzblatt).

Aufgabe 4: Korrektheitsbeweis *(6 Punkte)*

Sie erhalten folgendes Python-Programm:

```
1  a = [1, 2, 7, 8, 12, 34]
2  b = [1, 3, 7, 11, 17, 28]
3
4  result = []
5  i = 0
6  j = 0
7  for k in range(len(a) + len(b)):
8      if i == len(a):
9          result += b[j:]
10         j = len(b)
11     elif j == len(b):
12         result += a[i:]
13         i = len(a)
14     elif a[i] <= b[j]:
15         result += [a[i]]
16         i += 1
17     else:
18         result += [b[j]]
19         j += 1
```

Zeigen Sie mit einem totalen Korrektheitsbeweis, dass das Array `result` sortiert ist und die Elemente der Arrays `a` und `b` enthält.

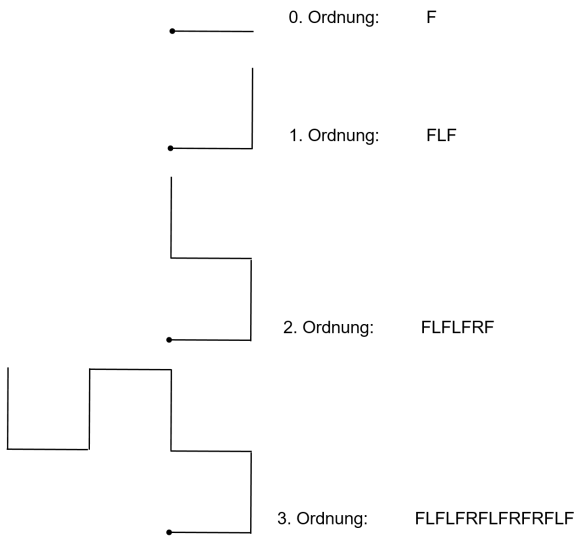
Aufgabe 5: Drachenkurve *(6 Punkte)*

Lesen Sie sich den Artikel über Drachenkurven auf Wikipedia durch.

Institut für Informatik
Scientific Computing and Bioinformatics
Prof. Dr. Andreas Hildebrandt
M.Sc. Jennifer Leclaire

Ihre Aufgabe ist es ein Programm zu schreiben, das die Anleitungen zum Zeichnen von Drachenkurven von $n = 0, 1, 2, ..., 5$ Ordnung ausgibt. Verwenden Sie dazu folgende Anweisungen als Strings:

- **F** (engl. forward) - Zeichne eine Einheit geradeaus.
- **L** (engl. left) - Drehe um 90° nach links.
- **R** (engl. right) - Drehe um 90° nach rechts.



Der Algorithmus zur Erstellung einer Drachenkurve n-ter Ordnung kann so definiert werden:

a) Zeichne eine Drachenkurve von $n-1$ -ter Ordnung.

b) Drehe um 90° nach links.

c) Zeichne eine Drachenkurve von $n-1$ -ter Ordnung in umgekehrter Reihenfolge, wobei (zusätzlich zur umgekehrten Reihenfolge) jede Anweisung **L** durch **R** und jede Anweisung **R** durch **L** ersetzt wird.