Institut für Numerische und Angewandte Mathematik
FB Mathematik und Informatik der Universität Münster
Prof. Dr. Mario Ohlberger
Dr. Christoph Lehrenfeld

14. März 2016

Übung zum Kompaktkurs

Einführung in die Programmierung zur Numerik mit Python

Wintersemesterferien 2015/2016 — Blatt 1

Aufgabe 1 (Wahrheitstabellen)

Schreiben Sie ein Programm, das für die folgenden Bool'schen Ausdrücke eine Wahrheitstabelle erstellt:

(a) $\text{nand}(a, b) \equiv \neg(a \wedge b)$

(b) $\text{xor}(a, b) \equiv (\neg a \wedge b) \vee (a \wedge \neg b)$

Eine Wahrheitstabelle gibt dabei für jede mögliche Belegung von a und b jeweils den Wert der Bool'schen Ausdrücke an. Beispielhaft für $a \wedge b$:

Belegung a	Belegung b	$a \wedge b$
w	w	w
w	f	f
f	w	f
f	f	f

Sie sollen die Wahrheitstabellen in einem `dictionary` speichern und auf dem Terminal ausgeben. Beachten Sie dazu, dass man wie folgt mehr-dimensionale dictionaries anlegen kann:

```
multi_dim_dict = {}
multi_dim_dict["key"] = {}
multi_dim_dict["key"]["another key"] = "an entry"
```

Zum Beispiel für den zweiten Eintrag aus obiger Tabelle:

```
my_dict[True][False] = True and False
```

Aufgabe 2 (Fibonacci-Zahlen)

Schreiben Sie ein Programm, das nach Eingabe einer natürlichen Zahl n die ersten n Fibonacci-Zahlen in einer Liste speichert. Die Fibonacci-Zahlen sind definiert durch

$$F_0 = F_1 = 1, \qquad F_n = F_{n-1} + F_{n-2} \text{ für } n \geq 2.$$

Geben Sie die Zahlen zusammen mit dem Quotienten $\frac{F_n}{F_{n-1}}$ (für $n \geq 1$) auf dem Terminal aus. Implementieren Sie auch eine Fehlermeldung für den Fall, dass eine negative Zahl eingegeben

worden ist. Testen Sie Ihr Programm für verschieden große Werte von n.

Hinweise:

- *In Python 2 liefert die Division zweier ganzen Zahlen wieder eine ganze Zahl. Initialisieren Sie die Fibonacci-Folge daher mit Gleitkommazahlen, d.h. mit* `1.0` *statt mit* `1`*. Sonst liefert der Quotient nicht die gewünschten Werte.*
- *Der Aufruf* `string = raw_input(message)` *(wobei* `message` *eine anzuzeigende Zeichenkette ist) speichert eine Benutzereingabe in der Variablen* `string`*. Das Ergebnis ist ein String, der z.B. mittels* `int(string)` *in eine ganze Zahl konvertiert werden kann.*
- *An eine Liste* `lst` *kann man mittels* `lst.append(value)` *einen Wert anhängen.*

Aufgabe 3 (Sieb des Eratosthenes)

Das Sieb des Eratosthenes ist eine Methode, mit der man eine Liste aller Primzahlen erstellen kann, die kleiner oder gleich einer gegebenen Schranke n sind. Dazu erstellt man zunächst eine Liste aller ganzen Zahlen $2, \ldots, n$ und entfernt nun Schritt für Schritt alle Vielfachen von 2, 3 usw. Implementieren Sie dieses Primzahlsieb. Verwenden Sie dabei als Datenstruktur für die Liste der ganzen Zahlen eine `set`. Lassen Sie die obere Schranke n vom Benutzer eingeben und schreiben Sie am Ende die Menge der gefundenen Primzahlen auf das Terminal.

Hinweise:

- *Der Ausdruck* `e in s` *liefert* `True`*, wenn das Element* `e` *in der Menge* `s` *enthalten ist, sonst* `False`*.*
- *Der Befehl* `s.remove(e)` *entfernt das Element* `e` *aus der Menge* `s`*.*