Programmierkurs Python I – WS 09/10

Die Probeklausur entspricht im Umfang *nicht* dem der Klausur: Die Klausur wird etwas umfangreicher ausfallen. Ausserdem wird die Klausur u.U. auch Aufgaben enthalten, die in ihrer Art von den Aufgaben in der Probeklausur etwas abweichen. Denkbar sind Aufgaben wie Fehler in einem Programm zu finden, oder vorgegebene Programme zu analysieren („Was berechnet folgende Funktion?“).

1 Summen und Rekursion

(a) Was versteht man unter *Rekursion* bzw. *rekursiven Funktionen*?

(b) Skizziere stichpunktartig einen (einfachen) rekursiven Algorithmus zur Berechnung der Summe der in einer Liste enthaltenen Zahlen. Alternativ kann der Algorithmus auch direkt in Python angegeben werden.

2 Lambda

Schreibe einen `lambda`-Ausdruck, der eine zweistellige Funktion (mit zwei Parametern `x` und `y`) darstellt, deren Rückgabewert eine Liste aller geraden Zahlen zwischen `x` und `y` ist.

```
>>> list(map(lambda ..., [1,3], [4,5]))
[[1, 2, 3, 4], [3, 4, 5]]
```

3 Variablenbindung

Zu welchem Wert wertet `outer(x)` aus, und welchen Wert hat `x` nach dem Aufruf von `outer(x)`?

```
x = 0

def outer(x):
    def inner():
        nonlocal x
        x += 1
```

```
        return x
    return inner()

outer(x)
```

4 List Comprehensions

Zu welchem Wert wertet die List-Comprehension aus?

```
zahlen = [1,2,3]
[(x,y) for x in zahlen for y in zahlen if x != y]
```

5 Verarbeiten von XML

Das folgende Dokument ist kein korrektes XML.

```
<a>
   <b>
      <c>Ein Text.</c>
      <c>Noch <d f=3>Noch ein Text.</c>
   </b>
</a>
<a>
    <b>Letzte Zeile<b>
</a>
```

Wie könnte man dieses Dokument mit Python verarbeiten, wenn man beispielsweise alle Text-Elemente, die unterhalb eines `c` Elements stehen, extrahieren möchte? Nenne kurz zwei Möglichkeiten. Du brauchst hier keinen Code anzugeben, es geht nur um die Benennung Technik.

6 Generatoren und Iteratoren

Was sind Generatoren, und was sind Iteratoren? Erkläre kurz beide Programmierkonzepte. Nenne Gemeinsamkeiten und Unterschiede bei der Anwendung und der Implementierung eigener Generatorn bzw. Iteratoren.

7 „Duck Typing“

Was versteht man unter „Duck Typing“? Nenne ein Beispiel aus der Vorlesung, bei dem dieses Konzept Anwendung findet.

8 Hooks

Methoden wie `__add__` oder `__eq__` nennt man auch *hooks* in Python. Wozu dienen diese Methoden? Wie und wofür kann man sie sich in eigenen Implementierungen zu Nutze machen?

9 Anonyme Funktionen

Was sind anonyme Funktionen? Nenne zwei Möglichkeiten, wie man sie in Python erzeugen kann.
