Programmierkurs Python I – WS 2010/11

Übung 3

---

1 Primzahlen mit `for` (1 Punkte)

In der Vorlesung wurde ein Algorithmus vorgestellt, der die Primzahlen in einem bestimmten Zahlenbereich berechnet und dafür eine `while`-Schleife benutzt.

Schreibe eine Funktion, die Primzahlen berechnet und dafür eine `for`-Schleife benutzt. Die Funktion soll als Parameter die Grenzen des Zahlenbereichs nehmen.

2 Fibonacci iterativ (3 Punkte)

Auf den Folien zur Vorlesung wird ein rekursiver Algorithmus für die Berechnung der Fibonacci-Zahlenfolge gezeigt. Implementiere eine *nicht-rekursive* Funktion, die die Fibonacci-Zahlen mit einer Schleife berechnet.

3 Listen sortieren (3 Punkte)

Der „Bubblesort“-Algorithmus ist ein einfacher (und wenig effizienter) Algorithmus, um eine gegebene Liste von Zahlen zu sortieren. Der Algorithmus geht wie folgt vor: Die Liste wird immer wieder „von links nach rechts“ durchlaufen; wenn zwei benachbarte Elemente in falscher Ordnung stehen, werden sie vertauscht. Der obige Schritt wird solange wiederholt, bis die Liste komplett sortiert ist.

Implementiere den Bubblesort-Algorithmus.

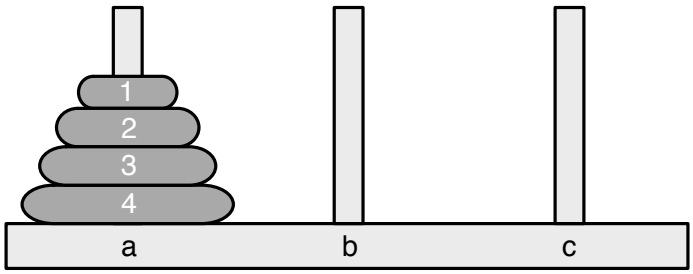
```
>>> bubblesort([5,2,1,4,6,3,1])
[1, 1, 2, 3, 4, 5, 6]
```

4 Default-Parameter (1 Punkte)

Erweitere die Bubblesort-Implementierung so, dass man beim Aufruf der Funktion angeben kann, ob die Liste auf- oder abwärts sortiert werden soll.

```
>>> bubblesort([5,2,1,4,6,3,1])
[1, 1, 2, 3, 4, 5, 6]
>>> bubblesort([5,2,1,4,6,3,1], reverse=True)
[6, 5, 4, 3, 2, 1, 1]
```

5 Die Türme von Hanoi (Bonus, 3 Punkte)



Die Türme von Hanoi sind ein berühmtes Beispiel für Probleme, die elegant rekursiv gelöst werden können. Der Spielaufbau besteht aus drei Stäben (im Bild a, b, c) und n Scheiben (im Bild n=4). Ziel ist es, den Scheibenturm von a auf einen anderen Stab zu bringen (z.B. c). Man darf jeweils nur eine Scheibe bewegen, und eine Scheibe darf nur auf eine größere Scheibe oder einen leeren Stab gelegt werden.

Implementiere den Algorithmus zur Lösung des Problems für n Scheiben. Eine Möglichkeit zur Implementierung besteht darin, die Spielzüge in der richtigen Reihenfolge mit `print` auszugeben (z.B. *Bewege Scheibe x von a nach c.*).

Hinweis: Um `n` Scheiben von a nach b zu bringen, bringe erst `n-1` Scheiben von a nach c, dann die `n`-te Scheibe nach b, und die `n-1` Scheiben von c nach b.

Abgabe bis Donnerstag, 11.11.09, 14.00 Uhr
per Mail an
`regneri@coli.uni-sb.de`
`stth@coli.uni-sb.de`