Einführung in die Programmierung

Prof. Dr. Detlef Krömker
Alina Stürck, M.Sc. Math.

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Übungsblatt Nr. 4[1]

Ausgabe: 16.11.2016
Abgabe: 03.12.2016
16:00 Uhr

Hotelmanagement

Hinweis: *Dieses Aufgabenblatt ist mit einer Bearbeitungszeit von zwei Wochen dafür ausgelegt in einem Zweierteam gelöst zu werden.*
Neben der Implementierung wird bei der Bearbeitung Wert auf die folgenden Punkte gelegt: Dokumentation[2], Strukturierung und Einhalten unseres Style-Guides[3].
Die genaue Aufschlüsselung der Bepunktung ergibt sich wie folgt:

- *50% entfallen auf die reine Funktionalität der Implementierung,*
- *25% für die Einhaltung des Style-Guides (insbesondere also auch für Dokumentation) und*
- *25% entfallen schließlich auf die angegebenen Testfälle.*

Achtung: Achten Sie darauf die Variable `__author__` in **allen** Quellcode Dateien **korrekt** zu setzen. Abgaben die nicht dieser Vorgabe entsprechen werden **nicht** bewertet! Quellcode muss als *.py* Datei und alles Weitere als *.pdf* Datei abgegeben werden. Bei Abgaben mehrer Dateien müssen diese als *.zip* zusammengefasst werden. Abgaben, die nicht diesen Regeln entsprechen, werden ebenfalls **nicht** bewertet! Außerdem muss ihr Name in jeder abgegebenen *.pdf* Datei zu finden sein.

Σ __ / **20**

Aufgabe 4.1: **Das Spiel**

Nehmen Sie an, Sie seien der Leiter eines erfolgreichen Hotels. Ihr Hotel ist sogar so erfolgreich, dass Sie von Ihrer Heimatstadt aus in andere Städte expandieren möchten. Diese Expansion folgt aber ganz bestimmten Regeln. Zur Erklärung werden die folgenden Variablen benötigt:

$5 \leq n \leq 20$	$n \in \mathbb{N}$	Anzahl der existierenden Städte
$5 \leq m \leq 20$	$m \in \mathbb{N}$	Anzahl der Manager in Ihrer Heimatstadt
$5 \leq d \leq 40$	$d \in \mathbb{N}$	Anzahl der verfügbaren Tage für die Expansion
$-20 \leq p_i \leq 90$	$p_i \in \mathbb{N}$	Möglicher Gewinn in Stadt i
$s_{k,l} \in \{0, 1\}$		Existenz einer Straße zwischen Stadt k und l

Zu Beginn des Spiels besitzen Sie 0 Geldeinheiten. Am Ende jedes Tages machen Sie in einer Stadt i, in der sich ein Hotel und mindestens ein Manager befinden, p_i Geldeinheiten Gewinn und in jeder Stadt mit Hotel ohne Manager 20 Geldeinheiten Verlust.

Ziel für den Spieler dieses Einzelspieler-Spiels ist es, am Ende der d Runden möglichst viele Geldeinheiten zu besitzen.

[1] Es dürfen keine Lösungen aus dem Skript, dem Internet oder anderen Quellen abgeschrieben werden. Diese Quellen dürfen nur mit Quellenangaben verwendet werden und es muss ein hinreichend großer Eigenanteil in den Lösungen deutlich zu erkennen sein.

[2] `http://www.python.org/dev/peps/pep-0257`

[3] `http://moodle.studiumdigitale.uni-frankfurt.de/moodle2/pluginfile.php/28852/mod_resource/content/1/Programmierhandbuch-2015.pdf`

Regeln: Zu Beginn des Spiels befinden sich m Manager in Ihrer Heimatstadt. Pro Stadt können Sie höchstens ein Hotel besitzen. Es besteht an jedem Tag die Möglichkeit genau eine der folgenden Aktionen auszuführen:

- Nichts tun
- Bauen(a): Baue ein neues Hotel in der Stadt a.
- Manager bewegen(x, a, b): Es werden x Manager von Stadt a zu einer benachbarten Stadt b bewegt.
- Neu Einstellen(a): Es wird ein neuer Manager in Stadt a angestellt, in der bereits ein Hotel existiert. Diese Aktion kostet 3 Tage.

Manager können sich nur über vorhandene Straßen ($s_{k,l} = 1$) bewegen.

Aufgabe 4.2: Implementierung

Punkte: ____ / 11

(a) (2 Punkte) **Spielrunden:** Implementieren Sie den groben Ablauf des Spiels, bei dem an jedem vergehenden Tag vom Benutzer Eingaben gemacht werden.

(b) (4 Punkte) **Startwerte:** Implementieren Sie eine Funktion, in welcher die oben genannten Startwerte $(n, m, d, p_i, s_{k,l})$ für das Spiel festgelegt werden. Diese sollen – je nach Wunsch des Benutzers – entweder durch Benutzereingaben spezifiziert werden oder zufällig erzeugt werden. Jede Stadt soll dabei einen eigenen Namen haben.

(c) (4 Punkte) **Aktionen:** Implementieren Sie die beschriebenen Aktionen, welche ein Spieler ausführen kann. Dabei sollen die Aktionen genau mit dieser Syntax beim Spielen verwenden werden können:

- Nichts tun: `pass`
- Bauen: `build: city`
- Manager bewegen: `move: number, city_1, city_2`
- Neu einstellen: `hire: city`

(d) (1 Punkt) **Gewinn:** Implementieren Sie die Berechnung des Gewinns an jedem Tag.

Aufgabe 4.3: User Interface

Punkte: ____ / 6

(a) (3 Punkte) Ermöglichen Sie es dem Benutzer jederzeit eine *help* Funktion aufzurufen, welche ihm die möglichen Eingaben anzeigt. Geben Sie ihm außerdem die Möglichkeit, das Spiel jederzeit zu beenden oder eine neue Runde zu starten.

(b) (2 Punkte) Nach jedem Zug soll die derzeitige Verteilung von Managern, Hotels, **möglichen** Gewinn p_i pro Stadt sowie den tatsächlichen Gewinn in der vorausgegangenen Runde angezeigt werden, z. B. in der Form

```
'Hamburg': Manager? Ja, Hotel? Nein, Mögl. Gewinn? 20, Tats. Gewinn? 0
'Dresden': Manager? Nein, Hotel? Nein, Mögl. Gewinn? -10, Tats. Gewinn? 0
```

oder

```
'Köln' (0, 1, 30, -20); 'Dresden' (0, 0, -10, 0); 'Berlin' (5, 1, 10, 10)
```

Außerdem soll eine Adjazenzmatrix der Straßen inklusive Namen der Städte ausgegeben werden, z. B. ei drei Städten von der Form

```
1          'Hamburg'  'Berlin'  'Köln'
2 'Hamburg'    1          0         0
3 'Berlin'     0          1         1
4 'Köln'       0          1         1
```

(c) (1 Punkt) Lassen Sie von Ihrem Spiel eine Highscore-Datei anlegen (dies darf z. B. eine *.txt* Datei sein), welche die 10 besten Ergebnisse (mit dem größten Gewinn) speichert. Am Ende jedes Spiels soll diese ausgelesen und das aktuelle Spielergebnis korrekt eingeordnet werden. Daraufhin sollen diese Top 10 Platzierungen ausgegeben und in der Datei abgespeichert werden.

Aufgabe 4.4: Fehlerbehandlung

Punkte: ____ / 3

Ihr *User Interface* soll robust angelegt sein, sodass falsche Eingaben des Benutzers **nicht** zu einem Absturz führen.

Hinweise:

Diese Hinweise sollen Ihnen eine Idee geben, welche Möglichkeiten Python bietet. Es ist natürlich nicht zwingend, die genannten Möglichkeiten zu nutzen.

- Graphen lassen sich gut mit sogenannten *Adjazenzmatrizen* repräsentieren. Unsere Straßen sind in beide Richtungen benutzbar, sodass eine Adjazenzmatrix immer symmetrisch wäre.
- Denken Sie auch über die Möglichkeiten zum Speichern ihrer Städte und der dazugehörigen Werte nach. Sie sind hier frei in Ihrer Wahl, jedoch sind nicht alle Datenstrukturen gleich gut geeignet.
- Zur Erzeugung von Zufallszahlen eignet sich das Modul `random`. Dort finden Sie beispielsweise die Funktion `random.choice(seq)`, welches ein zufälliges Element aus einer Sequenz `seq` auswählt und die Funktion `random.randint(a, b)`, welche einen zufälligen Integerwert $a \leq N \leq b$ auswählt.
- Dieser Zettel ist so konzipiert, dass sie auch einzelne Teilaufgaben machen können, wenn Sie nicht alles schaffen oder mal keine Idee haben ohne zu viele Punkte zu verlieren.