Übungen zu Computergrundlagen WS 2014/2015

Übungsblatt 7: Python IV

26.11.2014

Allgemeine Hinweise

- Abgabetermin für die Lösungen ist
  - **Freitag, 05.12.2014, 10:00**
- Schickt die Lösungen bitte per Email an Euren Tutor.

Aufgabe 7.1: Schreibe einen einfachen Sudoku-Löser in Python (10 Punkte)

Nachdem wir uns auf Blatt 4 Gedanken darüber gemacht haben, wie ein Sudoku (`http://en.wikipedia.org/wiki/Sudoku`) algorithmisch gelöst werden kann, soll es in dieser Aufgabe darum gehen, ein solches Programm in Python zu schreiben.

- **7.1.1** Lies das Sudoku-Puzzle aus der Datei `sudoku.dat` ein. Diese enthält 9×9 Zahlen zwischen 0 und 9 die den Inhalt der Sudokufelder von links oben nach rechts unten angeben. Die 0 steht dabei für ein leeres Feld. (1 Punkt)
- **7.1.2** Initialisiere eine geeignete Datenstruktur, um den Zustand des ungelösten Sudokus darzustellen. Dabei soll das Sudoku wie oben als Liste von 81 Elementen dargestellt werden, die den 81 Feldern von links oben nach rechts unten entsprechen. Dabei ist aber jedes Element jetzt eine Liste mit den möglichen Einträgen an dieser Stelle. Das heißt, dass Felder, die vorgegeben sind, als Eintrag eine Liste mit nur genau dieser vorgegebenen Ziffer bekommen, und alle anderen zunächst eine Liste mit Ziffern von 1 bis 9. (2 Punkte)
- **7.1.3** Schreibe drei Funktionen, die dir jeweils eine Zeile, Spalte oder $3x3$-Untereinheit des Sudokus zurückgeben. Die Funktionen sollten als Argumente ein Sudoku sowie Zeile und/oder Spalte der betreffenden Untereinheit annehmen und eine Liste mit 9 Feld-Elementen zurückgeben. (3 Punkte) **Hinweis:** Die Funktionsdeklarationen sollten also wie folgt aussehen:

```
def get_line(sudoku, column):         ...
def get_row(sudoku, row):             ...
def get_square(sudoku, column, row): ...
```

- **7.1.4** Löse das Sudoku nun, in dem du durch alle Felder gehst und überprüfst, welche Ziffern tatsächlich mögliche Einträge sind, und alle anderen entfernst. Überprüfe hierzu für jedes Element die zugehörige Zeile, Spalte und $3x3$-Untereinheit auf Zahlen, die bereits festgelegt sind. Ist in einem der Reihen, Spalten oder Unterquadrate eine Zahl bereits festgelegt, darf sie an keiner anderen Stelle mehr eingesetzt werden. Führe diesen Algorithmus solange durch, bis Du nichts mehr ausschliessen konntest, sich also nichts mehr ändern wird. (3 Punkte)
  Hinweis: Felder sind eindeutig zugewiesen, wenn die Liste für diese Felder genau *einen* Eintrag hat, also die Länge 1 hat. Beachte, dass dieser Algorithmus nur einfache Sudoku löst. Bei komplexeren Sudokus ist eine Technik namens *Backtracking* nötig, bei der man die verbleibenden Möglichkeiten durchprobiert. Das sprengt aber den Rahmen dieser Vorlesung.
- **7.1.5** Schreibe nun das gelöste Sudoku in eine Textdatei. Nutze hierbei dasselbe Format wie für die Eingabedatei.(1 Punkt)

Hinweise:
Hier noch einmal eine Wiederholung der wesentlichen Listenbefehle:

- Um zu prüfen, ob ein Element in einer Liste vorkommt, kannst du das `in`-Statement von Python verwenden:

```
numbers=[1, 3, 5, 8]
x=3
if x in numbers:
        print 'Element ', x, ' ist in Liste numbers'
```

- Elemente können mit der Listenfunktion `remove` entfernt werden:

```
        numbers.remove(x)
```

- Über die Elemente einer Liste kann man in Python einfach mit einer `for`-Schleife iterieren:

```
for num in numbers:
        print num, ' ist Element in Liste numbers'
```

- Denke daran, dass Python Listen normalerweise flach kopiert:

```
numbers = [range(1, 10), range(11, 20)]
flat = numbers[0]        # Flache Kopie von numbers
deep = numbers[0][:]     # Unabhaengige Kopie von numbers

flat.remove(3) # Hat Einfluss auf die Liste numbers,
               # entfernt auch dort die Ziffer 3
deep.remove(4) # Aendert die Liste numbers nicht
```

- Listen als Funktionsargumente werden flach kopiert, d. h. Änderungen der Liste schlagen auf die aufrufenden Ebenen durch.