Übungsblatt 4

Thomas Graf
EF / WF Informatik 2018-2019
Programmieren in Python II

13. September 2018

Tipp:
In gewissen der folgenden Aufgaben, werdet Ihr die Python-Funktionen für Listen, welche im Abschnitt *List Methods* in https://www.w3schools.com/python/python_lists.asp gegeben sind, verwenden müssen.

1 Weitere Beispiele von Listen ($\star$)

Wir haben bereits das Beispiel mit den, möglicherweise nicht-eindeutigen, E-Mail-Adressen gesehen. Überlege Dir mindestens drei weitere praktische Probleme, die sich mit Hilfe von Listen relativ einfach lösen lassen, ohne Listen aber nur mühsam zu lösen wären.
Wie würdest Du die Probleme aus deinen Beispielen lösen?

2 Liste von Namen sortieren ($\star$)

Erstelle eine Liste mit den Namen von mindestens fünf historischen Persönlichkeiten. Sortiere die Liste alphabetisch und gib sie aus. Füge einen weiteren Namen an das Ende der Liste an und sortiere erneut.

3 E-Mail-Adressen der KS Im Lee ($\star$)

Erkläre in Worten, wie man das Problem der möglicherweise nicht-eindeutigen E-Mail-Adressen aus dem Skript mit Hilfe der Python Funktion `count()` lösen könnte.

4 Grösstes Element in einer Liste (⋆⋆)

Erstelle eine Liste `number` von mindestens 8 beliebigen Zahlen. Schreibe danach (mit Hilfe eines for-loops) ein ganz simples Programm, welches die grösste Zahl in der Liste findet. Die Funktion `max` für Listen darf hier nicht verwendet werden.

5 Die zwei grössten Elemente in einer Liste (⋆⋆)

Wie könnte man das Programm aus 4 modifizieren, damit die **zwei** grössten Zahlen in der Liste gefunden werden?

6 Monotone Listen (⋆ ⋆ ⋆)

Eine Liste `L` heisst monoton ansteigend, falls für alle $i \leq j$, `L[i] <= L[j]`.
Eine Liste `L` heisst monoton fallend, falls für alle $i \leq j$, `L[i] >= L[j]`.
Eine Liste heisst monoton, falls sie entweder monoton ansteigend oder monoton fallend ist.
Schreibe eine Python-Funktion `def ist_monoton(L)`, welche `True` zurück gibt, falls `L` monoton ist und `False` sonst.

7 Wechselgeld (⚛)

Die Mensa der Universität Zürich (Mercato UZH Zentrum) verkauft am Mittag das Menü *de foifer*, welches für alle Kundinnen und Kunden genau 5 CHF kostet.
Nehmen wir an, dass dieses Menü nur an genau einer, speziell dafür vorgesehenen, Kasse bezahlt werden kann. Die Kundinnen und Kunden des *foifer*-Menüs stellen sich diszipliniert in einer Reihe an dieser Kasse an und bezahlen "eine Person nach der anderen".
Jede Person in der Reihe kauft genau ein *foifer*-Menü und bezahlt die 5 CHF entweder mit einem Fünflieber, einer Zehnernote oder einer 20-Franken-Note.

Die Reihenfolge der Bezahlungen sei in der Liste `payments` gespeichert.
Beispiel:

1. Person bezahlt mit einem Fünflieber
2. Person bezahlt mit einer Zehnernote
3. Person bezahlt mit einem Fünflieber

4. Person bezahlt mit einer 20-Franken-Note
5. ...

⇒ `payments = [5, 10, 5, 20, ... ]`

Die Kasse ist zu Beginn (bevor die erste Person bezahlt) leer.

Aufgabe:
Schreibe eine Python-Funktion

```
def wechselgeld(payments),
```

welche `True` zurück gibt, wenn bei gegebener Bezahl-Reihenfolge `payments` (Liste), jeder Person das korrekte Wechselgeld zurück bezahlt werden kann und `False`, falls dies nicht möglich ist.
Beispiel:

```
Input: payments = [5,5,5,10,20]
Output: True
Erklärung:
Von den ersten 3 Kundinnen / Kunden erhalten wir drei Fünflieber
Von der vierten Person erhalten wir eine Zehnernote und geben ihr ein
 ↪  Fünflieber zurück
Von der fünften Person erhalten wir eine 20-Franken-Note und geben ihr
 ↪  eine Zehnernote und ein Fünflieber zurück
Alle Personen erhalten das korrekte Wechselgeld -> True

Input: payments = [10,20]
Output: False
```

Abgabe: bis spätestens Montag, 17. September 2018, um 08:00.