Musterlösungen Übungsblatt 4

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

21. Oktober 2018

1 Weitere Beispiele von Listen ($\star$)

Es kommt sehr häufig vor, dass Programme irgendwelche Daten als Eingabe erhalten und versuchen eine gewisse Information aus diesen Daten herauszuholen (oder aus ihnen zu berechnen). Dabei ist es praktisch, wenn alle diese Daten in demselben Container abgespeichert sind. So ein Container könnte z.B. eine Liste sein.
Beispiele:

- Gegeben: Liste `visitors` mit den täglichen Besucherzahlen einer Webseite der letzten 20 Tage.
  Gesucht: durchschnittliche Anzahl der Besucher pro Tag.
  Vorgehen: berechne den Durchschnitt der
  Elemente der Liste
  `average = sum(visitors)/float(len(visitors)`

- Gegeben: Liste `swiss_sites` der Namen aller Webseiten mit der länderspezifischen Top-Level-Domain *.ch*.
  Frage: ist der Name *wieistmeineip* noch verfügbar.
  Vorgehen: die Frage ist offensichtlich äquivalent zur Frage, ob dieser Name in der gegeben Liste enthalten ist oder nicht:
  `verfuegbar = 'wieistmeineip' in swiss_sites`

- Gegen: Liste der Zeiten, welche die User auf einer Webseite verbracht haben.
  Gesucht: Durchschnitt der 5% kürzesten Zeiten, welche die User auf der Webseite waren.
  Vorgehen: ist der Leserin / dem Leser zur Übung überlassen.

2 Liste von Namen sortieren (⋆)

```
names = ['Claude Elwood Shannon', 'Alan Mathison Turing',
         'John von Neumann', 'Alonzo Church', 'Kenneth Lane Thompson']
names.sort() # sortiere die Namen alphabetisch
# fuege einen weiteren Namen hinzu
names.append('Dennis MacAlistair Ritchie')
print(names)
names.sort() # sortiere erneut
print(names)
```

3 Mail-Adressen der KS Im Lee (⋆)

Wir schreiben die Namen aller Schülerinnen und Schüler in eine Liste und geben ihr den Namen `students`. Nun gehen wir mit einer Schleife durch die Liste und rufen die Funktion `count` für jedes Element der Liste auf. Falls der Rückgabewert von `count` an mindestens einer Stelle grösser ist als 1, dann wissen wir, dass es mindestens zwei Schülerinnen oder Schüler mit demselben Namen an der Schule hat.

```
def is_unique_email(names):
    for name in names:
        if names.count(name) > 1:
            return(False)
    return(True)
```

4 Grösstes Element in einer Liste (⋆⋆)

```
def find_max(L):
    max = L[0]
    for element in L:
        if element > max:
            max = element
    return(max)

number = [34.12,-0.1,0.4,9,-100.2,88,0,2,-0.219]
print(find_max(number))
```

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

```
def find_max(L):
    max = L[0]
    for element in L:
        if element > max:
            max = element
    return(max)

def find_largest_two(L):
    first_max = find_max(L)
    L.remove(first_max)
    second_max = find_max(L)
    return([first_max,second_max])

number = [-0.1,0.4,88,9,-100.2,88,0,2,2,3,5,2]
print(find_largest_two(number))
```

6 Wechselgeld (⚛)

Vorgehen:
Wenn jemand mit einem Fünflieber bezahlt, dann nehmen wir ihn einfach.
Wenn jemand mit einer Zehnernote bezahlt, dann müssen wir genau einen Fünflieber zurück geben.
Wenn jemand mit einer 20-Franken-Note bezahlt, dann können wir entweder eine Zehnernote und ein Fünflieber zurück geben oder drei Fünflieber.
Die wichtigste Überlegung in der gesamten Aufgabe ist folgende:
Alle Bezahlungen, die mit einem Fünflieber und einer Zehnernote gemacht werden können, können gerade so gut mit drei Fünfliebern getätigt werden. Die Umkehrung gilt jedoch nicht:
Auf eine Bezahlreihnfolge der Form `[..., 10, 10, 10, ...]` kann mit einer Zehnernote und einem Fünflieber korrekte Wechselgeld geben werden, mit Fünfliebern getätigt werden, mit drei Fünfliebern jedoch schon. In diesem Sinne, sind die Möglichkeiten, mit einem Fünflieber und einer Zehnernote Wechselgeld zu geben, strikt schlechter als mit drei Fünfliebern. Deshalb werden wir immer zuerst versuchen, wenn möglich, das Wechselgeld auf eine 20-Franken-Note durch einen Fünflieber und eine Zehnernote zu geben.

Code:

```
def wechselgeld(payments):
    five = ten = 0
    for pay in payments:
        if pay == 5:
            five += 1
        elif pay == 10:
            if not five:
                return(False)
            five -= 1
            ten += 1
        else:
            if five and ten:
                five -= 1
                ten -= 1
            elif five >= 3:
                five -= 3
            else:
                return(False)
    return(True)
```