Übungen zu Computergrundlagen WS 2019/2020

Übungsblatt 9: Python I

20. Dezember 2019

Allgemeine Hinweise

- Abgabetermin für die Lösungen ist **Freitag, 10.01.2020, 11:00 Uhr**
- Schickt die Lösungen bitte per Email an Euren Tutor:
  - Montag 14:00–15:30: Moritz Schumacher (`mschumacher@icp.uni-stuttgart.de`)
  - Dienstag 9:45–11:15: Samuel Tovey (`stovey@icp.uni-stuttgart.de`)
  - Dienstag 15:45–17:15: Philipp Stärk (`pstaerk@icp.uni-stuttgart.de`)
  - Mittwoch 15:45–17:15: Marco Brückner (`mbrueckner@icp.uni-stuttgart.de`)
  - **geändert:** Donnerstag 9:45–11:15: Michael Kuron (`mkuron@icp.uni-stuttgart.de`)
- Die Übungen sollen von Gruppen von jeweils *zwei* (nur in Ausnahmefällen drei) Leuten bearbeitet werden. Bitte gebt *nur eine Lösung pro Gruppe* ab und nennt in eurer Abgabe alle Mitglieder eurer Gruppe!
- Als Lösung der Aufgabe soll ein einziges Python-Skript erstellt werden, welche ihr dann per E-Mail an euren Tutor schickt.

Aufgabe 9.1: Listenzugriff und -slicing (2 Punkte)

Gebt an, wie man in einer Python-Liste wie

```
a = [2, "d", 5, 8, 233, "dx", 54, "we", "g", ..., 72, 23, "g"]
```

auf folgende(s) Element(e) zugreifen kann. (2 Punkte)

- Das dritte Element
- Das zweite bis vierte Element
- Das letzte Element
- Jedes dritte Element, beginnend ab dem ersten
- Jedes zweite Element, beginnend ab dem zweiten
- Alle Elemente außer dem vorletzten
- Alle Elemente in umgekehrter Reihenfolge

Aufgabe 9.2: Datentypen (1 Punkt)

Sagt vorher, welches Ergebnis (Wert und Datentyp, oder eine Fehlermeldung) folgende Ausdrücke produzieren. Gebt jeweils den Grund dafür an. (1 Punkt)

- `3 + 5`
- `3 + 5.0`
- `"3" + "5"`
- `"3" * 5`
- `3//2`
- `3/2`
- `int(2.71828)`
- `round(2.71828)`
- `[1, 2] + [3, 4]`
- `"frohe"+ "Weihnachten"`
- `frohe + Weihnachten`

Aufgabe 9.3: Vergleich (2 Punkte)

Schreibt eine Funktion, die prüft, ob eine Liste sortiert ist. Verwendet folgendes Grundgerüst und prüft mit den Testfällen am Ende, ob eure Funktion korrekt arbeitet. (2 Punkte)

```
def is_sorted(data):
  ...
  for item in data:
    ...
    if ...:
      return False
    ...
  return True

assert is_sorted([1, 6, 9, 27])
assert is_sorted([1, 6, 6, 9, 27])
assert not is_sorted((3, 1, 4))
assert is_sorted(["Frohe", "Weihnachten"])
assert not is_sorted(["Frohes", "neues", "Jahr"])
```

Aufgabe 9.4: Primfaktorzerlegung (3 Punkte)

Schreibt eine Funktion, die eine Zahl in ihre Primfaktoren zerlegt und diese zurückgibt. Verwendet folgendes Grundgerüst und fügt einige Testfälle hinzu (wie bei der vorhergehenden Aufgabe), um zu prüfen, ob eure Funktion korrekt arbeitet. (3 Punkte)

```
def prime_factors(number):
  assert type(number) is int and number > 1
  factors = []
  while ...:
    ...
    ⋮
    ...
  return factors
```

Aufgabe 9.5: Histogramm (2 Punkte)

Ein Histogramm gibt die Wahrscheinlichkeitsverteilung einer kontinuierlichen Variable an. Dazu werden die Daten in sogenannte Klassen aufgeteilt, die jeweils alle Werte der Variable enthalten, die in einem bestimmten Intervall liegen. In diesem Beispiel nehmen wir die Geburtsdaten eines fiktiven Physikstudenten-Jahrganges (in der Datei `/group/cgl/2019/09/geburtsdaten.txt`) und assoziieren jedes Geburtsjahr mit einer Klasse. Vervollständigt das Grundgerüst unten, um zu bestimmen, welcher Anteil der Studenten in welchem Jahr geboren wurde. (2 Punkte)

```
histogram = {}

with open("geburtsdaten.txt") as f:
  for line in f:
    date = line.strip()
    year = ...date...
    ...
    ...
    histogram[year] = ...

for year in sorted(histogram.keys()):
  print(year, histogram[year])
```