Autor:
FTF

Einführung in die Programmierung
WS 2025/2026
Prof. Dr. Peter Thiemann

Gedächtnisklausur

19. Februar 2026

- Für die Bearbeitung der Aufgaben haben Sie **150 Minuten** Zeit.
- Es sind keine Hilfsmittel wie Skripte, Bücher, Notizen oder Taschenrechner erlaubt. Des Weiteren sind alle elektronischen Geräte (wie z.B. Handys) auszuschalten. Ausnahme: Fremdsprachige Wörterbücher sind erlaubt.
- **Falls Sie mehrere Lösungsansätze einer Aufgabe erarbeiten, markieren Sie deutlich, welcher gewertet werden soll**. Die geforderten Funktionen dürfen nur einmal in der Abgabe definiert werden, alles andere muss auskommentiert oder gelöscht werden.
- **Verwenden Sie Typannotationen**, um die Typen der Parameter und des Rückgabewertes Ihrer Funktionen anzugeben. Verwenden Sie Typvariablen, falls die Funktion für beliebige Typen gelten soll. Fehlende oder falsche Typannotationen führen zu Punktabzug.
- Bearbeiten Sie die einzelnen Aufgaben in den **vorgegebenen Musterdateien**, z.B. `ex3_strings.py`. Falsch benannte Funktionen werden nicht bewertet. Neu erstellte Dateien werden nicht bewertet.
- **Die Zielfunktionen dürfen ihre Eingaben nicht verändern**, d.h. Methoden wie `list.remove` dürfen nicht auf die Eingaben angewendet werden; es sei denn, die Aufgabenstellung fordert explizit die Eingabe zu verändern.
- Intern darf Ihre Implementierung den vollen Sprachumfang verwenden; **es sei denn, die Aufgabenstellung schließt etwas aus**.
- **Sie dürfen keine Module importieren**. Alle Imports, die benutzt werden dürfen/müssen sind bereits vorgegeben. Zum Lösen der Aufgaben sind keine weiteren Importe/Module notwendig.

Es sind maximal 150 Punkte zu erreichen.

Aufgabe 1 (Warm-Up; 20 Punkte)

(a) (? Punkte) Definieren Sie eine Funktion `my_len()`, die einen string `a` als Argument entgegennimmt und die länge von `a` zurückgibt.

Sie dürfen die Built-In Methode len(str) *NICHT* verwenden.

(b) (? Punkte) Implementieren sie die Funktion `fac` die eine natürliche Zahl `n` als Argument entgegennimmt und ihre Fakultät zurückgibt.

Die Fakultät sei definiert als:

$$\text{fac}(n) := \begin{cases} 1 \text{ für } n = 0 \\ n \cdot \text{fac}(n-1) \text{ für } n > 0 \end{cases}$$

Hinweis: Sie dürfen annehmen, dass n größer gleich 0 ist.

(c) (? Punkte) Definieren Sie einen Typalias MyOptional, der einen Typparameter `T` besitzt und einen Wert beschreibt, der entweder vom Typ `T` oder None ist.

(d) (? Punkte) Gegeben sei die rekursive Definition eines Bin¨arbaumes in Python:

```python
@dataclass
class Node[T]:
  mark: T
  left: "Tree[T]"
  right: "Tree[T]"

type Tree[T] = Optional[Node[T]]
```

Ändern sie diese Definition so, dass der Baum eine beliebige, endliche Anzahl an Kindern annehmen kann.

(a) (? Punkte) Implementieren sie eine Funktion die ein Iterierbares Objekt annimmt und einen Iterator zurückgibt, der die gleichen Elemente des Eingabeobjektes in der gleichen Reihenfolge wiedergibt.

Aufgabe 2 (Dictionaries; 20? Punkte)
Diese Aufgabe soll Universitätskurse simulieren. Es seien zwei Dictionaries gegeben:

- `courses_rooms` ist ein Dictionary, das zu jedem Kurs (String) als Key den jeweiligen Raum (String) als Value zuordnet.
- `courses_students` ist ein Dictionary, das zu jedem Kurs (String) als Key die Menge an Studierenden (string) als Value hat.

(a) (? Punkte) Implementieren Sie eine Funktion, die `courses_rooms` und `courses_students` annimmt und diejenigen Kurse in `courses_rooms` zurückgibt, die nicht in `courses_students` sind.

(b) (? Punkte) Implementieren Sie eine Funktion, die `courses_students` , eine Menge an Studierenden `graduates` sowie eine Mindestmenge `min_students` annimmt. Sie soll das Dictionary `courses_students` *inline* ändern, sodass diejenigen Studierenden entfernt werden, die in `graduates` sind. Die Funktion soll zudem die Menge an Kursen zurückgeben, in denen sich weniger als `min_students` Studierende befinden.

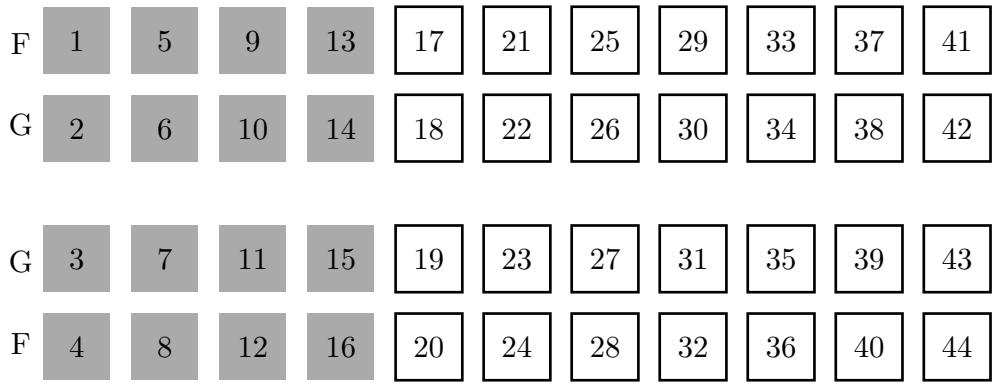
(c) (? Punkte) Implementieren Sie eine Funktion, die `courses_rooms` und `courses_students` sowie eine Mindestmenge `min_students` annimmt. Die Funktion gibt ein Dictionary zurück, die allen Räumen die Menge derjenigen Kursen zuordnet, in denen sich mindetens `min_students` befindent.

Aufgabe 3 (Strings; 10 Punkte)
Diese Aufgabe efasst sich mit CamelCase und snake_case. Diese Formatierungsarten sind so definiert:

- snake_case: String sind im snake_case Format, wenn es nur aus Kleinbuchstaben und Unterstrichen besteht. Wörter sind mit einem Unterstrich getrennt.
- CamelCase: String sind im CamelCase Format, wenn Wörter zusammengeschrieben sind und, außer im ersten Wort, der erste Buchstabe des Worte großgeschrieben ist, der Rest kleingeschrieben.

(a) (5 Punke) Implementieren Sie eine Funktion `snake_to_camel`, die einen String im snake_case Format annimmt und diesen Sting in CamelCase format umwandelt und zurückgibt.
(b) (5 Punke) Implementieren Sie eine Funktion `camel_to_snake`, die einen String im CamelCase Format annimmt und diesen Sting in snake_case format umwandelt und zurückgibt.

Aufgabe 4 (Datenklassen; ? Punkte)
bla bla keine dups mit erbung



(a) Implementieren sie die Datenklasse `Seat` der die Attribute Sitznummer `num`, welcher Name (String) den Sitz reserviert hat oder None `reversed` und den Wahrheitswert `window`, welcher sperichert, ob der Sitz ein **F**enstersitz wie im Obrigen Diagramm zu sehen ist.

`reserved` soll den Standartwert `None` haben und `window` soll automatisch bei erstellung eines `Seat` anhand von `num` bestimmt werden.

(b) Implementieren sie die Datenklasse `FirstClassSeat`, welche von `Seat` erbt und den Wahrheitswert `desk` speichert, welcher aussagt ob der Sitz einen Tisch hat.

Alle Sitze in der ersten Klasse mit num <=8 haben Tische.

First Class Seat Seat(desk bool) num < 8

(a) Feedback booked alreadyBooked, invalidSeatNumber
(b) Carriage `__seat_list num_first_class_seats num_second_class_seats`
- %4=0, >=0, list = []
- get_by_name
- get_by_number
- get_all_free_seats
- book -> Feedback

Aufgabe 5 (Endrekursion; ? Punkte)
bin(0)=„“ bin(2n)=bin(n)+„0“ bin(2n+1)=bin(n)+„1“

(a) (? Punkte) Schreiben Sie eine **rekursive, aber nicht endrekursive** Funktion `to_bin_rec`

(b) (? Punkte) Schreiben Sie eine endrekursive Funktion `to_bin_tail_rec`, die sich wie `sum_list_rec` verhält, aber endrekursiv implementiert ist. Verwenden Sie hierzu das in der Vorlesung gezeigte Verfahren mit einem Akkumulator acc als zusätzlichem Argument.

(c) (? Punkte) Schreiben Sie eine nicht-rekursive Funktion `to_bin_iter`, die sich wie `to_bin_rec` verhält, aber endrekursiv implementiert ist. Verwenden Sie hierzu das in der Vorlesung gezeigte Verfahren mit einem Akkumulator acc als zusätzlichem Argument.

Aufgabe 6 (Mobile und Arsch; ? Punkte)

```python
@dataclass
class Arm[T]:
  weight: int
  left: "Tree[T]" = None
  right: "Tree[T]" = None

type Mobile[T] = Optional[Arm[T]]
```

(a) (5 Punkte) is balanced -> tuple (bool, sum_weight)
(b) (10 Punkte) is_isomorphic -> bool

Aufgabe 7 (Generatoren; 20 Punkte)

(a) sum null

(b) encoding

(c) waren das nur 2?

Aufgabe 8 (Funktionale Programmierung; 20 Punkte)

(a) rev_app `#wtf` f(x)

(b) fehlt hier noch eine ?

(c) isprime(n) -> bool

(d) primes(xs: Iterable) -> dict(n, isprime(n))

(e) pipe: fs: list[callable], start