EINFÜHRUNG IN DIE PROGRAMMIERUNG

EXAM PREPARATION

DHBW MANNHEIM
WIRTSCHAFTSINFORMATIK (DATA SCIENCE)

Markus Menth
Martin Gropp

HINWEISE

Die folgenden Aufgaben dienen der Vorbereitung auf die Klausur. Sie sind jedoch nicht gleichzusetzen mit einer möglichen Klausur und dienen nur als Vorbereitungshilfe.

TEIL 1

AUFGABE 1 (15 PUNKTE)

```python
def f(x, y, z):
    x = x + 1
    y + [1, 2, 3]
    z.append(1)

def g(x):
    def h(y):
        return x * y
    return h

def k(x, z=1, **kwargs):
    return f'x={x}, z={z}, kwargs={kwargs}'


def k2(x):
    return x(5)

w = 9
x = 1
y = [5, "hallo"]
z = ["x"]
```

```python
f(x, y, z)
i = g(x)
j = g(10)

print(" 1.) ", x)
print(" 2.) ", y)
print(f" 3.) {z}")
print(" 4.) ", i(5))
print(" 5.) ", j(5))
print(" 6.)", w > 5)
print(" 7.)", w % 4)
print(" 7.)", w / 4)
print(" 8.)", w // 4)
print(" 9.)", w > 10 or w < 0)
print(" 9.)", w << 2)
print(" 9.)", w >> 2)
print("10.)", w & 0b101011001)
print("11.)", k(5, a=5, u=987))
print("12.)", k2(lambda i: i+5))
```

Stellen Sie die Ausgabe des Programms dar.

AUFGABE 2

Schreiben Sie eine Funktion `reverse` die einen als Parameter übergebenen String rückwärts zurückgibt. Dies muss über eine List Comprehension gelöst werden. Rufen Sie die Funktion mit dem String `Hallo Welt` auf.

AUFGABE 3

Geben Sie an, was im folgenden Code-Beispiel ausgegeben wird, und beschreiben Sie, wie es dazu kommt!

```
a = ['a', 'b', 'c']
b = a

b.append('d')
print(a[-1])
```

Worin besteht der Zusammenhang zu diesem Code-Fragment?

```
def f(x):
    x.append(10)

l = [1, 2, 3]
f(l)
print(l)
```

AUFGABE 4

Beschreiben Sie die Funktionsweise einer Closure anhand eines konkreten Beispiels.

AUFGABE 5

Beschreiben Sie das Problem der Seiteneffekte anhand eines Beispiels. Hinweis: nutzen sie dabei das Schlüsselwort `global`.

TEIL 2

AUFGABE 1 (15 PUNKTE)

Implementieren Sie ein Programm, das das kleine Einmaleins (1er-Reihe, die 2er-Reihe, die 3er- Reihe usw.) bis zur 10er-Reihe in folgender Form ausgibt:

```
1x1 = 1
2x1 = 2
...
9x10 = 90
10x10 = 100
```

AUFGABE 2 (15 PUNKTE)

Wortpalindrome ergeben vorwärts und rückwärts gelesen dasselbe Wort (z.B. `Rentner`).

Implementieren Sie eine Funktion in Python

- Dieser muss ein Wort übergeben werden
- Die Funktion gibt `True` zurück, wenn es sich um ein Wortpalindrom handelt, sonst `False`.
- Das Wort kann in Groß- oder Kleinschreibung übergeben werden, diese ist zu ignorieren.

AUFGABE 3 (20 PUNKTE)

Narzisstische Zahlen sind natürliche Zahlen, die durch die Anwendung bestimmter Rechenvorschriften auf ihre Ziffern sich selbst ergeben. Eine Armstrong-Zahl ist eine narzisstische Zahl, deren Summe ihrer Ziffern, jeweils potenziert mit der Stellenanzahl der Zahl, wieder die Zahl selbst ergibt.

Implementieren Sie eine Funktion in Python, der eine Zahl übergeben werden kann, die `True` zurückgibt, wenn es sich hierbei um eine Armstrong-Zahl handelt und `False`, wenn nicht.

Der Funktion werden ausschließlich positive Ganzzahlen übergeben. Eine Prüfung auf fehlerhafte Werte ist nicht zu implementieren.

AUFGABE 4 (15 PUNKTE)

Gegeben sei eine unsortierte Liste beliebiger Länge aus Ganzzahlen. Implementieren Sie ein Programm, das mithilfe der Listen-Abstraktion (list comprehension) eine neue Liste erzeugt, die nur Werte aus der ursprünglichen Liste enthält, die restlos durch 7 teilbar sind.

AUFGABE 5 (35 PUNKTE)

Die binäre Suche ermöglicht es, in sortierten Listen sehr effizient ein bestimmtes Element zu finden. Der Ablauf der binären Suche ist folgendermaßen:

- Im ersten Schritt wird das mittlere Element der Liste mit dem zu findenden Wert verglichen.
- Ist es kleiner als das gesuchte Element, muss sich das gesuchte Element in der ersten Hälfte der Liste befinden.
- Ist es größer, muss es sich in der zweiten Hälfte befinden.
- Ist es gleich dem gesuchten Element, ist die Suche beendet.

Beispiel: Suche nach 22 in der Liste

Schritt 0

- Sortierte Liste liegt vor

Indizes	0	1	2	3	4	5	6	7
Werte	**3**	**5**	**8**	**15**	**17**	**18**	**22**	**25**

Schritt 1

- Der Wert in der Mitte ist 15
- 15 ist kleiner als 22 → gehe in die zweite Hälfte der Gesamtliste

Indizes	0	1	2	3	4	5	6	7
Werte	3	5	8	15	**17**	**18**	**22**	**25**

Schritt 2

- Der Wert in der Mitte ist 18
- 18 ist kleiner als 22 → gehe in die zweite Hälfte zweiten Hälfte

Indizes	0	1	2	3	4	5	6	7
Werte	3	5	8	15	17	18	**22**	**25**

Schritt 3

- Der Wert in der Mitte ist 22
- 22 ist der gesuchte Wert &rarr& gib Index 6 zurück

Schreiben Sie eine rekursive Funktion, der eine Liste mit Werten und ein Suchwert übergeben werden kann und die den Index des Suchwertes zurückgibt.

Für die Lösung der Aufgabe gelten folgende Annahmen:

- Die übergebene Liste enthält nur Ganzzahlen
- Die übergebene Liste ist aufsteigend sortiert
- Die Anzahl der Elemente in der Liste kann gerade oder ungerade sein
- Bei einer geraden Anzahl an Elementen ist egal, bei welchem der mittleren Elemente die Suche beginnt
- Wird nach einem Wert gesucht, der in der Liste mehrfach vorkommt, ist egal, welcher der zugehörigen Indizes zurückgegeben wird
- Der zu suchende Wert befindet sich in der Liste

AUFGABE 7 (PUNKTE)

Implementieren Sie eine Klasse `Rectangle` (= Rechteck).

- Die `__init__`-Methode verlangt die Höhe und die Breite und setzt die übergebenen Werte in Attribute
- Eine Methode `get_area` der Klasse `Rectangle` gibt dessen Fläche zurück

Implementieren Sie zudem eine Klasse `Square` (= Quadrat), die sich von Rechteck ableitet

- `Square` hat keine eigenen Attribute oder Methoden (außer einer `__init__`-Methode).
- Der `__init__`-Methode von `Square` wird nur ein Parameter `width` übergeben, sie gibt aber Werte für Höhe und Breite an `Rectangle` weiter.