

# Python (OOP)

## Prüfungsvorbereitung

# Aufgabe 1

```
1 class Fantasy:
2
3     n = 0
4
5     def __init__(self, x):
6         self.x = x
7         Fantasy.n += 1
8
9     def add(self, other):
10        return Fantasy(self.x + other.x)
11
12    def minus_one(self):
13        self.x = self.x - 1
14
15    def size():
16        return Fantasy.n
```

# Aufgabe 1

```
18 a = Fantasy(4)
19 b = Fantasy(3)
20 c = a.add(b)
21 b.minus_one()
22
23 print(a.x)
24 print(b.x)
25 print(c.x)
26 print(Fantasy.size())
```

- (a) Welche Ausgabe(n) macht das folgende Python-Modul?
- (b) Welche Eigenshafte(n) hat ein Fantasy-Objekt?
- (c) Welche Eigenshafte(n) hat die Fantasy-Klasse?
- (d) Welche Methode(n) hat ein Fantasy-Objekt?
- (e) Welche Methode(n) hat die Fantasy-Klasse?

# Aufgabe 1

(a) Welche Ausgabe(n) macht das folgende Python-Modul?

# Aufgabe 1

(a) Welche Ausgabe(n) macht das folgende Python-Modul?

4

2

7

3

(b) Welche Eigenschaft(e)n hat ein Fantasy-Objekt?

# Aufgabe 1

(a) Welche Ausgabe(n) macht das folgende Python-Modul?

4

2

7

3

(b) Welche Eigenschafte(n) hat ein Fantasy-Objekt?

x

# Aufgabe 1

(a) Welche Ausgabe(n) macht das folgende Python-Modul?

4

2

7

3

(b) Welche Eigenshafte(n) hat ein Fantasy-Objekt?

x

(c) Welche Eigenshafte(n) hat die Fantasy-Klasse?



# Aufgabe 1

(a) Welche Ausgabe(n) macht das folgende Python-Modul?

4

2

7

3

(b) Welche Eigenschafte(n) hat ein Fantasy-Objekt?

x

(c) Welche Eigenschafte(n) hat die Fantasy-Klasse?

n

# Aufgabe 1

(a) Welche Ausgabe(n) macht das folgende Python-Modul?

4

2

7

3

(b) Welche Eigenshafte(n) hat ein Fantasy-Objekt?

x

(c) Welche Eigenshafte(n) hat die Fantasy-Klasse?

n

(d) Welche Methode(n) hat ein Fantasy-Objekt?

# Aufgabe 1

(a) Welche Ausgabe(n) macht das folgende Python-Modul?

4

2

7

3

(b) Welche Eigenshafte(n) hat ein Fantasy-Objekt?

x

(c) Welche Eigenshafte(n) hat die Fantasy-Klasse?

n

(d) Welche Methode(n) hat ein Fantasy-Objekt?

add, minus\_one

# Aufgabe 1

(a) Welche Ausgabe(n) macht das folgende Python-Modul?

4

2

7

3

(b) Welche Eigenshafte(n) hat ein Fantasy-Objekt?

x

(c) Welche Eigenshafte(n) hat die Fantasy-Klasse?

n

(d) Welche Methode(n) hat ein Fantasy-Objekt?

add, minus\_one

(e) Welche Methode(n) hat die Fantasy-Klasse?

# Aufgabe 1

(a) Welche Ausgabe(n) macht das folgende Python-Modul?

4

2

7

3

(b) Welche Eigenshafte(n) hat ein Fantasy-Objekt?

x

(c) Welche Eigenshafte(n) hat die Fantasy-Klasse?

n

(d) Welche Methode(n) hat ein Fantasy-Objekt?

add, minus\_one

(e) Welche Methode(n) hat die Fantasy-Klasse?

size

# Aufgabe 1

(a) Welche Ausgabe(n) macht das folgende Python-Modul?

4

2

7

3

(b) Welche Eigenshafte(n) hat ein Fantasy-Objekt?

x

(c) Welche Eigenshafte(n) hat die Fantasy-Klasse?

n

(d) Welche Methode(n) hat ein Fantasy-Objekt?

add, minus\_one

(e) Welche Methode(n) hat die Fantasy-Klasse?

size

## Aufgabe 2

Welche Ausgabe macht das folgende Programm?

```
1 class A:
2
3     c = 1
4
5     def __init__(self, a):
6         self.a = a
7
8     def __str__(self):
9         return 'a={0}'.format(self.a)
10
11    def calc(self, x):
12        return self.a * x
```

## Aufgabe 2

```
14 class B:
15
16     c = 10
17
18     def __init__(self, a):
19         self.a = a
20
21     def __str__(self):
22         return 'a={0}'.format(self.a)
23
24     def calc(self, x):
25         return self.a + x
```



## Aufgabe 2

```
27 d = A(4)
28 e = B(5)
29
30 print(d)
31 print(e)
32 print(A.c + B.c)
33 print(d.calc(2))
34 print(e.calc(2))
```

## Aufgabe 2

a=4

a=5

11

8

7

## Aufgabe 3

Welche Ausgabe macht das folgende Programm?

```
1 class Personal:
2
3     L = []
4
5     def __init__(self, name, abteilung, lohn):
6         self.name = name
7         self.abt = abteilung
8         self.lohn = lohn # pro Monat
9         Personal.L.append(self)
10
11     def __str__(self):
12         return '{0.name}/{0.abt}/{0.lohn}'.format(self)
13
14     def lohnerhoehung(self, erhoehung):
15         self.lohn += erhoehung
```

## Aufgabe 3

```
17     def liste():
18         return [self.name for self in Personal.L]
19
20 p1 = Personal('Lisa Meier', 'Marketing', 7000)
21 p2 = Personal('Sam Klein', 'Produktion', 6000)
22 p3 = Personal('Liz Hauser', 'Entwicklung', 8000)
```

## Aufgabe 3

Sam Klein/Produktion/6500

['Lisa Meier', 'Sam Klein', 'Liz Hauser']

## Aufgabe 4

Welche Ausgabe macht das folgende Programm, das die Definition einer Klasse für Schwarzweissbilder mit  $3 \times 4$  Bildpunkten enthält.  
*Hinweis:* 'W' steht für *White* und 'B' für *Black*.

```
1 class Image:
2
3     def __init__(self):
4         self.width = 4
5         self.height = 3
6         self.data = [['W', 'W', 'W', 'W'],
7                       ['W', 'W', 'W', 'W'],
8                       ['W', 'W', 'W', 'W']]
9
10    def set_pixel(self, row, column, color):
11        self.data[row][column] = color
12
13    def show(self):
14        for row in range(0, self.height):
15            print(''.join(self.data[row]))
```

## Aufgabe 4

```
17 img = Image()  
18 img.set_pixel(0, 0, 'B')  
19 img.set_pixel(1, 1, 'B')  
20 img.set_pixel(2, 2, 'B')  
21 img.set_pixel(1, 3, 'B')  
22 img.show()
```

# Aufgabe 4

BWW

WBWB

WWBW



## Aufgabe 5

- (a) Welche Ausgabe macht das folgende Programm?
- (b) Vervollständige die Methode `verkauf`, welche aufgrund eines Verkaufs den aktuellen Bestand eines Artikels um `anzahl` Einheiten verkleinert.

```
1 class Quartierladen:
2
3     def __init__(self):
4         self.sortiment = {'brot': 0, 'orangen': 0}
5
6     def neu(self, artikel, anzahl):
7         self.sortiment[artikel] = anzahl
8
9     def einkauf(self, artikel, anzahl):
10        self.sortiment[artikel] += anzahl
```

## Aufgabe 5

```
12     def verkauf(self, artikel, anzahl):
13         ...
14
15     def inventar(self):
16         for art in self.sortiment:
17             print('{0} {1}'.format(art,
18                                     self.sortiment[art]))
18
19 q = Quartierladen()
20 q.einkauf('brot', 10)
21 q.neu('milch', 7)
22 q.einkauf('brot', 2)
23 q.inventar()
```

# Aufgabe 5

## Aufgabe 5

brot 12  
orangen 0  
milch 7

## Aufgabe 6

```
1 class Schulfach:
2
3     def __init__(self, name):
4         '''Objekt mit name und leerer Notenliste'''
5         ...
6
7     def neue_note(self, note):
8         '''Fügt eine neue Note der Notenliste hinzu'''
9         ...
10
11    def get_note(self, i):
12        '''Gibt die i-te Note zurück'''
13        ...
14
15    def durchschnitt(self):
16        '''Gibt den Notendurchschnitt zurück'''
17        ...
```

Gegeben ist der Klassentwurf für die Notenverwaltung eines Schulfachs.

(a) Schreibe Client-Code, der die Klasse wie folgt *verwendet*:

- ▶ Erzeuge ein Schulfach-Objekt mit dem Namen 'Informatik' und weise es dem Bezeichner `inf` zu.
- ▶ Füge der Reihe nach die Noten 3, 4 und 5 hinzu.
- ▶ Gib den Wert der zweiten Note aus.
- ▶ Gib den Durchschnitt aller Noten aus.

(b) Implementiere die Methoden.

Beachte, dass es hier keine Rolle spielt, wie die „internen“ Variablen heissen und wie die Methoden genau programmiert sind, so lange das Resultat den Vorgaben im Docstring ('''...''') entspricht.

## Aufgabe 6

```
20 # (a) Client Code
21 inf = Schulfach('Informatik')
22 inf.neue_note(3)
23 inf.neue_note(4)
24 inf.neue_note(5)
25 print(inf.get_note(1))
26 print(inf.durchschnitt())
```

```
1 class Schulfach:
2
3     def __init__(self, name):
4         '''Objekt mit name und leerer Notenliste'''
5         self.name = name # (b)
6         self.L = []      # (b)
7
8     def neue_note(self, note):
9         '''Fügt eine neue Note der Notenliste hinzu'''
10        self.L.append(note) # (b)
11
12    def get_note(self, i):
13        '''Gibt die i-te Note zurück'''
14        return self.L[i] # (b)
15
16    def durchschnitt(self):
17        '''Gibt den Notendurchschnitt zurück'''
18        return sum(self.L)/len(self.L) # (b)
```



## Aufgabe 7

Vervollständige den Konstruktor und die übrigen Methoden der Klasse `Dreieck` gemäss den Docstrings. Der Flächeninhalt eines Dreiecks kann aus den drei Seiten mit der Flächenformel von Heron berechnet werden. Mit dem halben Umfang  $s = (a + b + c)/2$  des Dreiecks berechnen wir  $A = \sqrt{s(s - a)(s - b)(s - c)}$ .

Die Wurzel eines Ausdrucks (...) kann entweder mit der importierten Wurzelfunktion `sqrt(...)` oder wegen  $\sqrt{x} = x^{\frac{1}{2}}$  durch Potenzieren mit dem Exponenten 0.5 berechnet werden.

```
1 from math import sqrt
2
3 class Dreieck:
4
5     def __init__(self, a, b, c):
6         '''Erzeugt ein Dreieck-Objekt'''
7         ...
8
9     def __str__(self):
10        '''Gibt eine Textdarstellung der Seiten zurück.'''
11        ...
12
13    def umfang(self):
14        '''Gibt den Umfang des Dreiecks zurück.'''
15        ...
16
17    def inhalt(self):
18        '''Gibt den Flächeninhalt des Dreiecks zurück.'''
19        ...
```

## Aufgabe 7

```
1 from math import sqrt
2 class Dreieck:
3
4     def __init__(self, a, b, c):
5         '''Erzeugt ein Dreieck-Objekt'''
6         self.a, self.b, self.c = a, b, c # Platzgründe!
7
8     def __str__(self):
9         '''Gibt eine Textdarstellung der Seiten zurück.'''
10        return 'a={0.a}, b={0.b}, c={0.c}'.format(self)
11
12    def umfang(self):
13        '''Gibt den Umfang des Dreiecks zurück.'''
14        return a + b + c
15
16    def inhalt(self):
17        '''Gibt den Flächeninhalt des Dreiecks zurück.'''
18        s = (a + b + c)/2
19        return sqrt(s*(s-a)*(s-b)*(s-c))
```

## Aufgabe 8

Beschreibe die Grundidee der objektorientierten Programmierung und nenne zwei Vorteile, die sich daraus ergeben.

## Aufgabe 8

Die objektorientierte Programmierung ist ein Modell, das sich beim Schreiben eines Programms an der Realität des zu lösenden Problems ausrichtet. Die Objekte sind die Akteure, welche Aufgaben erledigen, ihren Zustand mitteilen/ändern und mit anderen Objekten kommunizieren können.

Solche Programme (1) sind besser verständlich/lesbar (2) sind leichter zu pflegen/warten (3) sind wiederverwendbar und (4) eignen sich für die Programmierung im Team.

## Aufgabe 9

Beschreibe die folgenden Fachausdrücke aus dem OOP-Bereich.

- (a) Klasse
- (b) Objekt (Instanz)
- (c) Objektvariable (Objekteigenschaft)
- (d) Klassenvariable (Klasseneigenschaft)
- (e) Objektmethode
- (f) Klassenmethode
- (g) Konstruktor
- (h) Kapselung

## Aufgabe 9

- (a) *Klasse*: Bauplan für Objekte eines bestimmten Typs
- (b) *Instanz*: Ein zur Laufzeit erzeugtes Objekt einer Klasse.
- (c) *Objektvariable (Objekteigenschaft)*: Variable, die an ein Objekt gebunden ist.
- (d) *Klassenvariable (Klasseneigenschaft)*: Variable, die an eine Klasse gebunden ist.
- (e) *Objektmethode*: Funktion, die an ein Objekt gebunden ist.
- (f) *Klassenmethode*: Funktion, die an eine Klasse gebunden ist.
- (g) *Konstruktor*: Methode (Funktion), die beim Erzeugen eines Objekts aufgerufen wird.
- (h) *Kapselung*: Das Prinzip, die Implementierung (also die Realisierung) einer Klasse nach aussen hin zu verbergen und den Zugriff über dafür vorgesehene Methoden (Schnittstelle) zu steuern.