Klasse Zutat. Folgender

4. (3 Punkte) Schreibe Code, der den folgenden Dialog möglich macht:

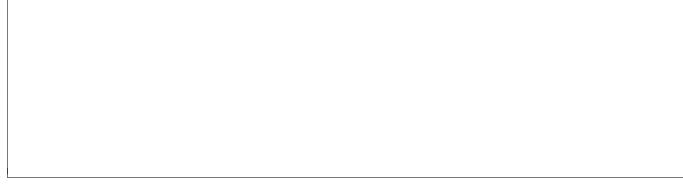
```
\begin{array}{l} k1 \, = \, K(1) \\ k2 \, = \, K(2) \\ \textbf{print} \, (k1.\, x \, , \ k1.\, y \, , \ k2.\, x \, , \ k2.\, y) \end{array}
```

 $1 \ 1 \ 2 \ 2$ 



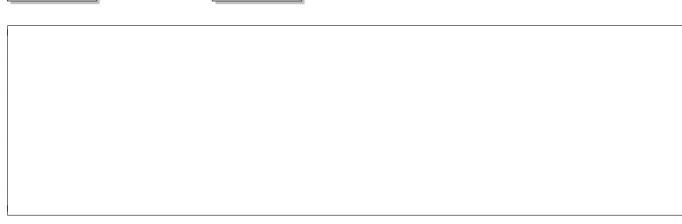
5. (4 Punkte) Die Klasse A hat ein Attribut zahl. Dem Konstruktor von A wird der Wert für das Attribut zahl übergeben. Die Klasse B erbt von A und erhält ein weiteres Attribut zahl2. Dem Konstruktor von B werden Werte für beide Attribute übergeben, der Konstruktor von B nutzt den Konstruktor von A. Implementiere die Klasse B.





6. (3 Punkte) Die Klasse A hat eine Methode berechne, die einen int-Wert zurückgibt. Die Klasse B erbt von A und überschreibt die Methode berechne. Sie addiert noch 10 auf das Ergebnis von A drauf. Implementiere die Methode berechne in der Klasse B.





'				nes Kontos wird der Wisprinzips mit getter u	
Implementie	ere die Klasse Person		Geheimnisprinzips mi	Person wird der Wert wit getter und setter-Mesein soll.	
Implementie	ere die Klasse Person	unter Wahrung des	Geheimnisprinzips mi	it getter und setter-Me	

- 9. (6 Punkte) a. Gib für die beiden Klassen ein passendes Klassendiagramm an.
  - b. Was erscheint auf der Konsole?

```
class L:
    def __init___(self,a,d):
        self.a = a
        self.d = d

def m1(self):
        self.a = self.d.x + 1

def m2(self):
        return self.a

class P:
    def __init___(self,x):
        self.x = x

p = P(1)
k = L(2,p)
print(k.m1())
print(k.m2())
```

- 10. (6 Punkte) a. Gib für die Klasse ein passendes Klassendiagramm an.
  - b. Was erscheint auf der Konsole?

```
class K:
     def = \inf_{s \in If , k1} (self, s):
            self.b1 = False
            self.s \,=\, s
      def m1(self, b2):
            if b2:
                 \textbf{return} \ \ \texttt{self.s} \ + \ \textbf{'}\#\textbf{'}
            else:
                  \textbf{return} \ \texttt{self.s} + \ \texttt{'?'}
      def bla(self):
           return not self.b1
      def m3(self, k):
            self.k1 += k
            \mathtt{self.b1} = \mathbf{not} \ \mathtt{self.b1}
     def __str__(self):
    return str(self.k1) + ' ' + str(self.b1) + ' ' + self.s
k = K('hi')
k.m3(4)
temp = k.m1(k.bla())
k\,.\,s \;=\; temp
print(k)
```

- 11. (8 Punkte) a. Implementiere eine Klasse Buch mit den Attributen autor und titel.
  - b. Erzeuge zwei Buch-Objekte b1 und b2: Homo faber von Max Frisch und Buddenbrooks von Thomas Mann.
  - c. Erstelle eine Liste buecher die b1 und b2 enthält.
  - d. Erstelle ein Dictionary buchmap mit dem Titel eines Buches als key und dem Buch-Objekt als value. buchmap soll zu Beginn Einträge für b1 und b2 enthalten.
  - e. Erweitere buchmap um einen Eintrag: Agnes von Peter Stamm.
  - f. In buchmap seinen noch mehrere Bücher eingefügt worden. Welcher Ausdruck mit buchmap liefert den Autor von Der Prozess (wir nehmen an, dass das entsprechende Buch in buchmap vorhanden ist).

12. (9 Punkte) a. Implementiere eine Klasse Filmmusik mit den Attributen film und komponist, so dass die folgenden Filmmusik-Objekte erzeugt werden können:

```
m1 = Filmmusik("Jurassic Park", "John Williams")
m2 = Filmmusik("Star Wars", "John Williams")
m3 = Filmmusik("Terminal", "John Williams")
m4 = Filmmusik("Inception", "Hans Zimmer")
m5 = Filmmusik("Interstellar", "Hans Zimmer")
m6 = Filmmusik("Lord of the Rings", "Howard Shore")
```

b. Implementiere eine string-Methode, die mit print (m1) folgende Ausgabe erzeugt:

Jurassic Park - Musik: John Williams

- c. Erzeuge eine Liste fmlist mit den Objekten m1, m2... m6. Die Objekte m1 bis m6 können als erzeugt vorausgesetzt werden
- d. Gehe mit einer Schleife durch die fmlist und gib alle Filmtitel mit Musik von John Williams aus.
- e. Erzeuge für die 6 Filme in fmlist ein Dictionary fmmap, bei dem jedem Komponistennamen (key) eine Liste seiner Filmmusik-Objekte als value zugeordnet ist.

Beispiel einer Nutzung von fmmap:

```
>>> for f in fmmap["Hans Zimmer"]:
    print(f)

Inception - Musik: Hans Zimmer
Interstellar - Musik: Hans Zimmer
```

13. (8 Punkte) Welche Ausgabe erscheint jeweils auf der Konsole?

```
class Clock:
     \mathbf{def} __init__(self, time):
            \overline{self}.\overline{time} = time
     \begin{array}{c} \textbf{def} \ \text{print\_time(self):} \\ \text{time} = \text{`6:30'} \end{array}
           \mathbf{print} (self.time)
clock = Clock(,5:30,)
clock.print time()
class Clock:
     \mathbf{def} __init__(self, time):
           self.time = time
      def print_time(self, time):
           print (time)
clock = Clock('5:30')
clock.print_time('10:30')
class Clock:
     def __init__(self, time):
    self.time = time
     def print_time(self):
    print(self.time)
boston_clock = Clock('5:30')
paris_clock = boston_clock
paris\_clock.time = , \overline{1}0:30,
boston_clock.print_time()
```

14.	(3 Punkte) Es sollen Objekte aus dem Bereich der Vektorgeometrie modelliert werden. Eine Ebene, eine Gerade und ein Punkt. Ein Punkt soll durch drei float-Werte x1, x2, x3 festgelegt sein. Eine Gerade soll durch zwei Punkte
	bestimmt sein, eine Ebene soll durch eine Gerade (ihre Normale) und einen Punkt definiert sein. Zeichne ein passendes Klassendiagramm mit den Klassennamen und den Attributen.
15.	(3 Punkte) Es sollen Objekte aus dem Bereich der Vektorgeometrie modelliert werden. Eine Ebene, eine Gerade und ein Punkt. Ein Punkt soll durch drei float-Werte x1, x2, x3 festgelegt sein. Eine Gerade soll durch zwei Punkte p1 und p2 bestimmt sein, eine Ebene soll durch 3 Punkte a, b, c definiert sein. Zeichne ein passendes Klassendiagramm mit den Klassennamen und den Attributen.