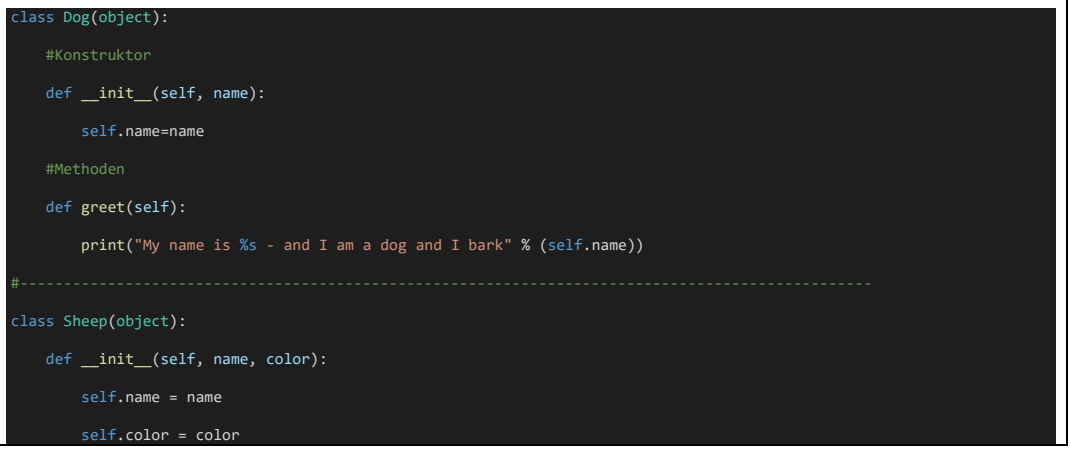
AAP10: Python – Klassen & Vererbung

task	Was ist eine Klasse? Was zeichnet Klassen aus? Wie werden diese in Python kodiert? Was ist ein Objekt? Was ist Vererbung? Wo/Wie hilft Vererbung?
Datum:	20.12.2019
Autor:	Plunser Fabio
Dokumentation:	Unterricht
Was Klassen passen für unsere Zootiere? Welche Eigenschaften hat ein Tier? - Membervariablen Was kann ein Tier? - Methoden	Ja Objekt classen? Eigenschaften: Einen Sound machen und hallo sagen
Erkläre am Animal-Beispiel die Begriffe Klasse und Objekt. Was wird in einer Klasse festgelegt? Wie definiere ich diese in Python? Was ist ein Objekt? Wie erzeuge ich dieses in Python?	Der class Befehl bestimmt in Python eine Klasse Es wird festgelegt mit def was dieses Objekt kann.
Ergänze den gemeinsam entwickelten Zoo um eine Schlangenklasse. Diese soll zusätzlich Beißen können – bite()! Was passiert, wenn ich in der FOR Schleife alle Tiere beißen lasse? Wann tritt welcher Fehler auf? Kann ich den abfangen?	Wenn die anderen Klassen die Methode Bite() nicht haben entsteht eine Exception.
Dokumentiere den Programmstand im Arbeitsauftrag in Form von Code als Text, Code als screenshot und screenshot der Ausgabe zur Laufzeit im Anhang!	

```python
class Dog(object):

    #Konstruktor

    def __init__(self, name):

        self.name=name

    #Methoden

    def greet(self):

        print("My name is %s - and I am a dog and I bark" % (self.name))

#-----------------------------------------------------------------------------------------

class Sheep(object):

    def __init__(self, name, color):

        self.name = name

        self.color = color
```

	`` def greet(self): print("My name is %s - and I am eating grass"%(self.name)) def printcolor(self): print("Ich bin %s"%(self.color)) #-- class Cow(object): def __init__(self, name): self.name=name def greet(self): print("My name is %s - MUUUUUUUUUH"%(self.name)) #-- class Snake(object): def __init__(self, name): self.name=name def greet(self): print("I'm a Snake an my name is %s"%(self.name)) def bite(self): print("I can bite") #-- def main(): Hasso = Dog(name = "Hasso") Wolli= Dog(name = "Wolli") sheep1= Sheep(name ="Dolli", color="Blau") schlange = Snake(name = "Hallo") animals=[Hasso, Wolli, sheep1] for a in animals: a.greet() a.printcolor() schlange.bite() if __name__ == '__main__': main() ```
Vererbung: Was stört den guten Programmieren am Programm? Wie kann ich das verbessern? Erkläre das Konzept der Vererbung mit Vater und Kindklassen. Was wird dadurch einfacher?	Vererbung, ein Objekt übernimmt alle definitions die ein drüberstehendes Objekt bereits hat. Man muss es nur einmal schreiben. Es wird einfacher Methoden weiter zu geben (man muss sie nicht mehr mitkopieren)
Die super()-Methode – Was kann ich damit erreichen? Wie rufe ich die Konstruktor-Methode der	Die super()-Methode hilf irgendwie auf die Methoden der Vaterklasse zu zugreifen. Genauer hab is nit verstanden.

Vaterklasse auf?	
Schreibe Dein Programm so um, dass es eine Animal-Vaterklasse und für jede Tierart eine Kindklasse gibt. Dokumentiere das neue Programm wieder wie gehabt!	

```
class Animal(object):
    def __init__(self, name="unbekannt"):

        self.name=name

    def greet(self):
        print("My name is %s - and I am a %s and I %s"
              % (self.name,self.species(),self.sound()))

    def species(self):
        typ=str(type(self))
        print(typ)
        return(typ[17:-2])
class Dog(Animal):
    def __init__(self, name="Bello"):
        super().__init__(name)
        self.color="brown"
    def sound(self):
        return("bark")
class Sheep(Animal):
    def __init__(self, name="Shaun"):
        super().__init__(name)
        self.color="brown"
    def sound(self):
        return("mäh")
class Snake(Animal):
    def __init__(self, name="Kaa"):
        super().__init__(name)
        self.color="brown"
    def sound(self):
        return("ssss")
    def bite(self):
        print("u ded")
if __name__ == '__main__':
    mein_hund=Dog(name="Hasso")
    mein_hund.greet()

    bello=Dog("Bello")
    dolly=Sheep("Dolly")
    dolly.greet()
    shaun=Sheep();
    hasso=Dog(name="Hasso")
    bello=Dog("Bello")
    kaa=Snake("Kaa")
    kaa.greet()
    kaa.bite()

    # einzelne Tiere stellen sich vor
    shaun.greet()
    bello.greet()
```

```
# und nun einige der Reihe nach
print("\nDie Objekte können der Reihe nach in einem Feld stehen")
animals=[shaun, dolly, hasso]
for a in animals:
    a.greet()

# Auf das erste Feldelement zugreifen - ->shaun
animals[0].greet()

# Schleife weniger elegant formuliert - C-Style
print("\nund nochmals alle ausgeben")
for i in range(len(animals)):
    animals[i].greet()
```

Abgabe des Arbeitsauftrags als PDF im moodle als AAP10-YourName.pdf	

Screenshots / documentation of your Classes, Objects / source code: