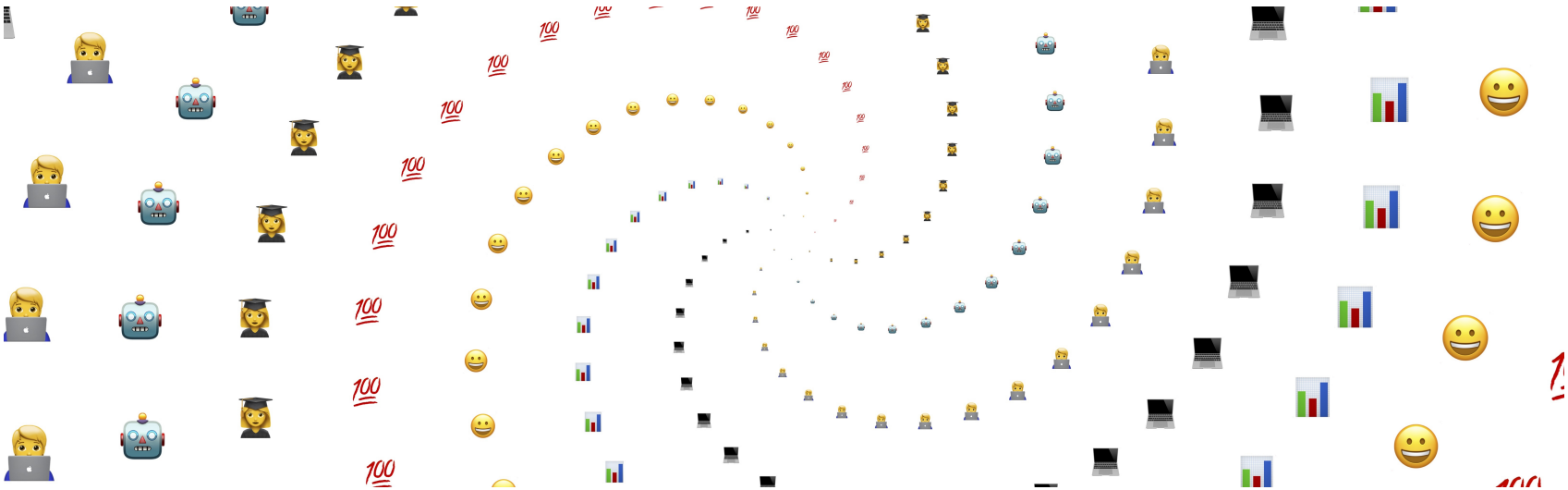
Programmieren: Klassen

Klassen

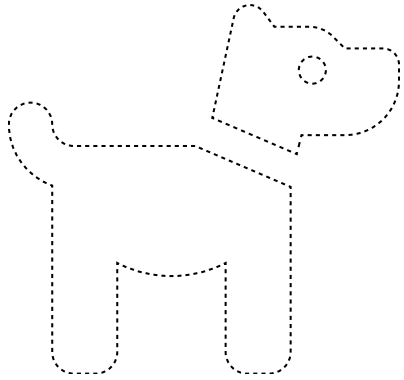
Cyril Wendl

Fachschaft Informatik
Kantonsschule im Lee



Klassen in Python

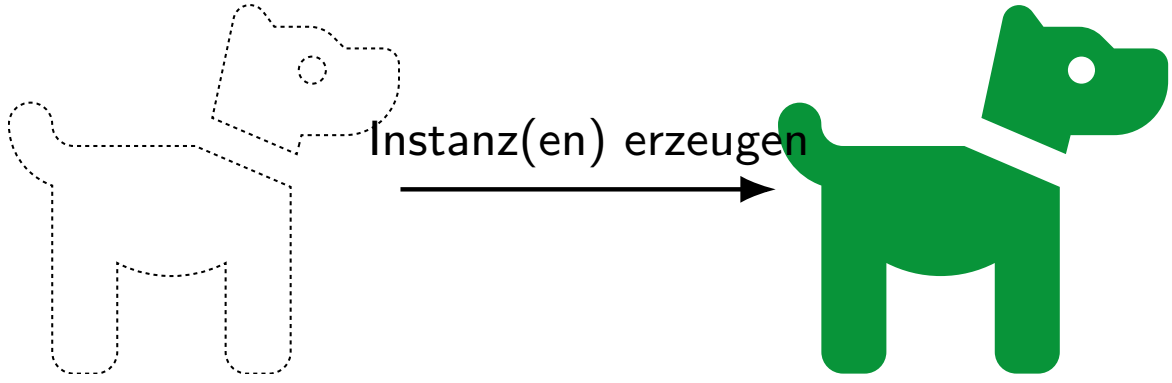
Klasse



```
class Hund:
```

Klassen in Python

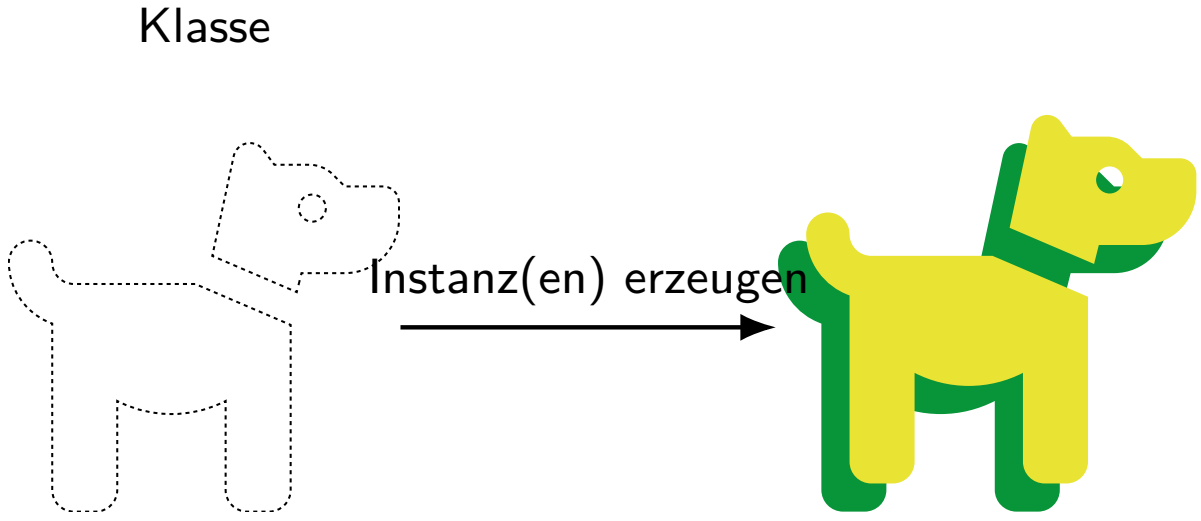
Klasse




```
class Hund:
```

```
hund1 = Hund("Thommy", 510, "Grün")
hund2 = Hund("Bello", 380, "Gelb")
hund3 = Hund("Rocky", 315, "Rot")
# ...
```

Klassen in Python

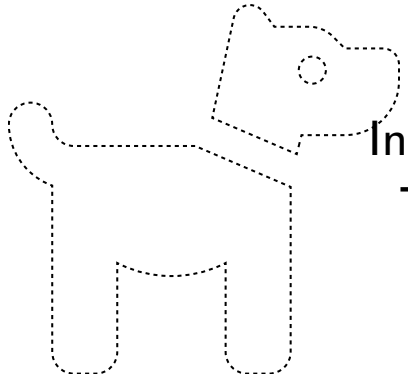



```
class Hund:
```

```
hund1 = Hund("Thommy", 510, "Grün")
hund2 = Hund("Bello", 380, "Gelb")
hund3 = Hund("Rocky", 315, "Rot")
# ...
```

Klassen in Python

Klasse



Instanz(en) erzeugen

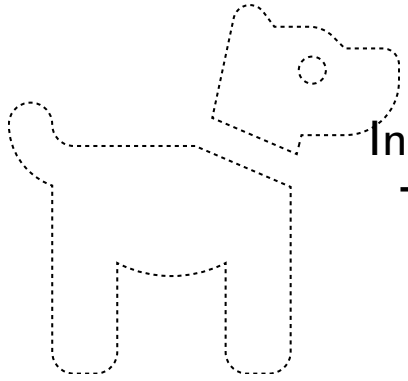
```
class Hund:
```

```
hund1 = Hund("Thommy", 510, "Grün")
hund2 = Hund("Bello", 380, "Gelb")
hund3 = Hund("Rocky", 315, "Rot")
# ...
```

Klassen in Python

Klasse

Objekte / Instanzen



Instanz(en) erzeugen

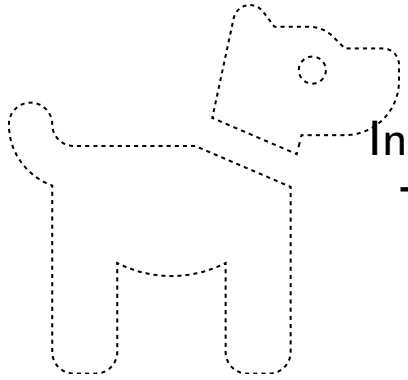
```
class Hund:
```

```
hund1 = Hund("Thommy", 510, "Grün")
hund2 = Hund("Bello", 380, "Gelb")
hund3 = Hund("Rocky", 315, "Rot")
# ...
```

Klassen in Python

Klasse

Objekte / Instanzen



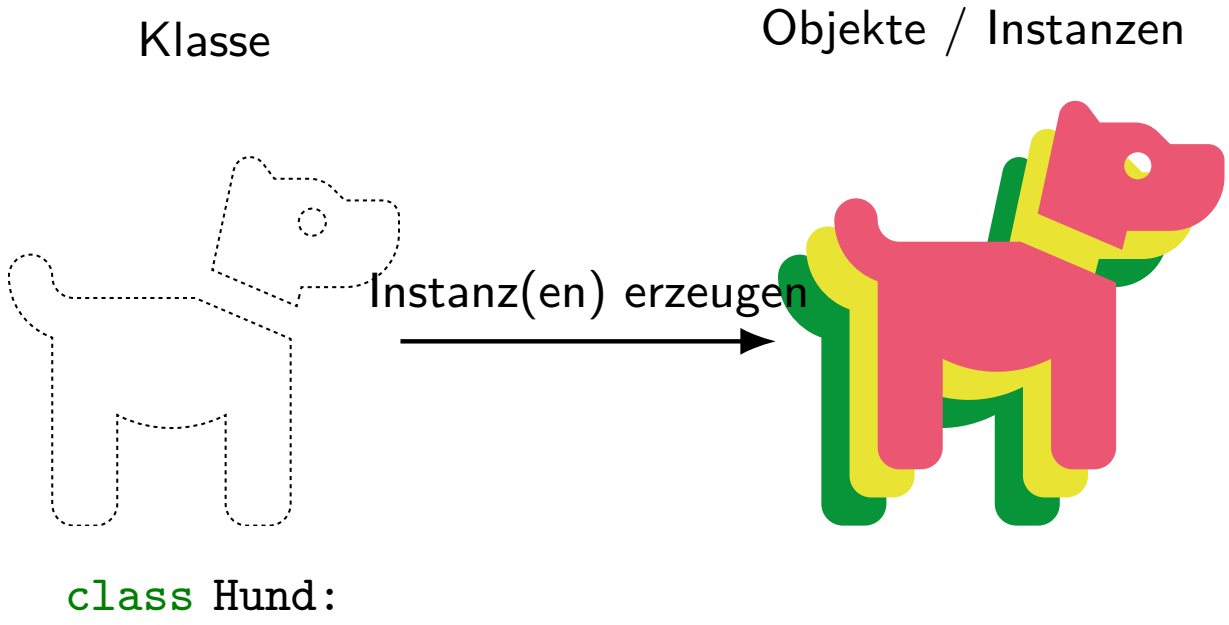
Instanz(en) erzeugen

```
class Hund:
```

```
hund1 = Hund("Thommy", 510, "Grün")
hund2 = Hund("Bello", 380, "Gelb")
hund3 = Hund("Rocky", 315, "Rot")
# ...
```

- Eigenschaften / Attribute: `name`, `farbe`, `alter`, etc.

Klassen in Python

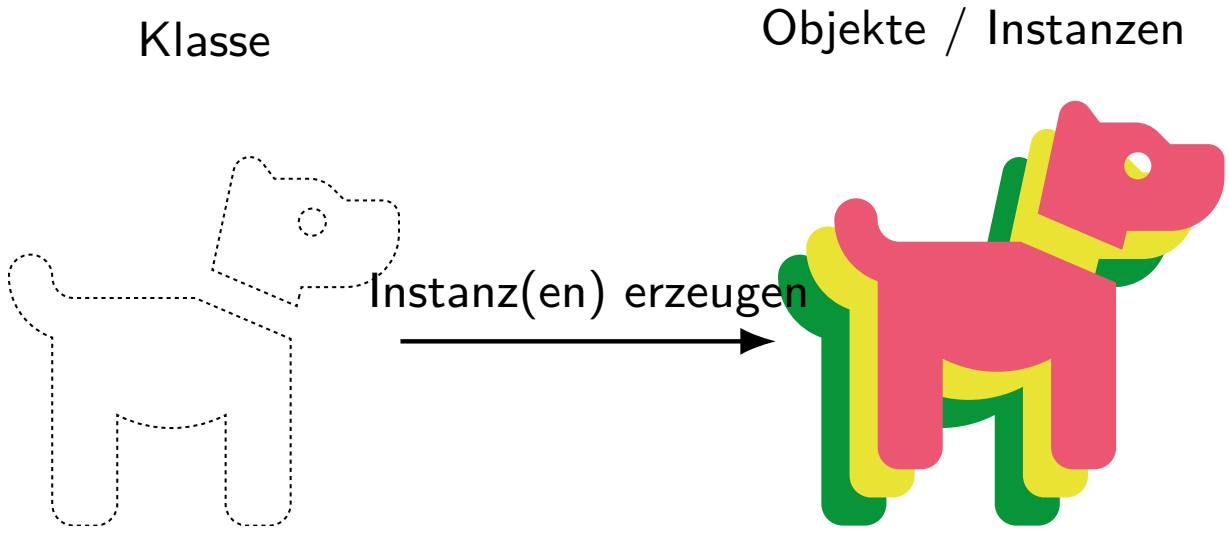



```
class Hund:
```

```
hund1 = Hund("Thommy", 510, "Grün")
hund2 = Hund("Bello", 380, "Gelb")
hund3 = Hund("Rocky", 315, "Rot")
# ...
```

- Eigenschaften / Attribute: `name`, `farbe`, `alter`, etc.
- Methoden: `.bellen()`, `.sitzen()`, etc.

Klassen in Python




```
class Hund:
```

```
hund1 = Hund("Thommy", 510, "Grün")
hund2 = Hund("Bello", 380, "Gelb")
hund3 = Hund("Rocky", 315, "Rot")
# ...
```

- Eigenschaften / Attribute: `name`, `farbe`, `alter`, etc.
- Methoden: `.bellen()`, `.sitzen()`, etc.
- **Wird in Games ständig verwendet, um Objekte zu „spawnen“!**

Klassen in Python

- Eine Klasse ist ein Bauplan für Objekte.

Klassen in Python

- Eine Klasse ist ein Bauplan für Objekte.
- Sie definiert Eigenschaften (Attribute) und Verhalten (Methoden) von Objekten.

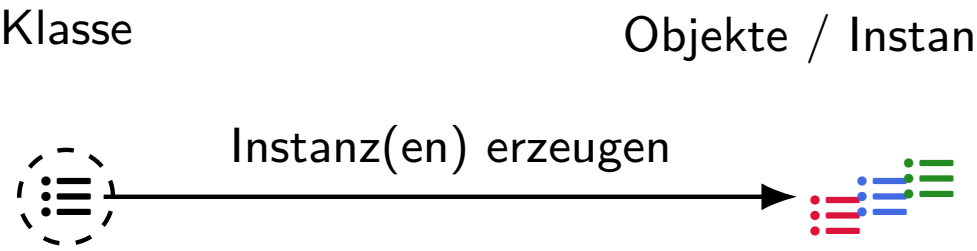
Klassen in Python

- Eine Klasse ist ein Bauplan für Objekte.
- Sie definiert Eigenschaften (Attribute) und Verhalten (Methoden) von Objekten.
- Klassen ermöglichen die Erstellung von benutzerdefinierten Datentypen.

Klassen in Python

- Eine Klasse ist ein Bauplan für Objekte.
- Sie definiert Eigenschaften (Attribute) und Verhalten (Methoden) von Objekten.
- Klassen ermöglichen die Erstellung von benutzerdefinierten Datentypen.
- Wir kennen Klassen bereits von anderen Typen: **Listen, Zahlen, Zeichenketten!**

Listen als Klassen




```
class list:
```

Attribute
Elemente
Länge
...

Methoden
append()
remove()
sort()
pop()
count()
...

```
liste1 = [1,2,3]
liste2 = ["a","b"]
liste3 = [True, False]
# ...
```

Attribute
Elemente : [1,2,3]
Länge : 3
...

Methoden
append()
remove()
sort()
pop()
count()
...

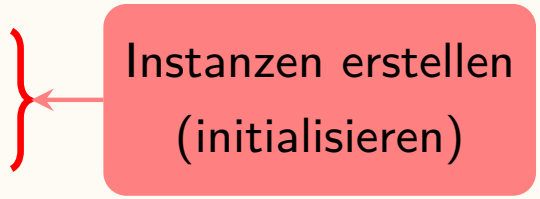
Beispiel: Personen

```python
class Person:
    def __init__(self, name, alter):
        self.name = name
        self.alter = alter

    def vorstellen(self):
        print(f"Hallo, ich bin {self.name} und {self.alter} Jahre alt.")

    def geburtstag_feiern(self):
        self.alter += 1
        print(f"{self.name} ist jetzt {self.alter} Jahre alt.")



# Objekte der Klasse Person erstellen
anna = Person("Anna", 16)
jan = Person("Jan", 17)

# Methoden aufrufen
anna.vorstellen()  # Ausgabe: Hallo, ich bin Anna und 16 Jahre alt.
jan.vorstellen()   # Ausgabe: Hallo, ich bin Jan und 17 Jahre alt.

# Geburtstag feiern
anna.geburtstag_feiern()  # Ausgabe: Anna ist jetzt 17 Jahre alt.
```

`class` ... **ist der Klassenname,** `self` **bezieht sich auf die aktuelle Instanz.**

Beispiel: Personen

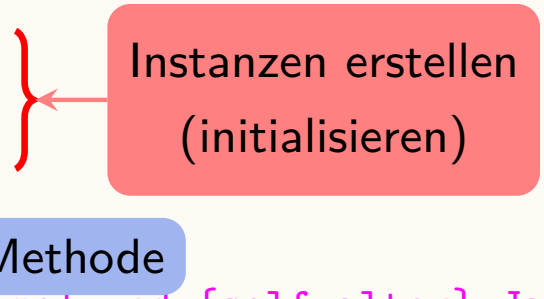
```python
class Person:
    def __init__(self, name, alter):
        self.name = name
        self.alter = alter

    def vorstellen(self):
        print(f"Hallo, ich bin {self.name} und {self.alter} Jahre alt.")

    def geburtstag_feiern(self):
        self.alter += 1
        print(f"{self.name} ist jetzt {self.alter} Jahre alt.")



# Objekte der Klasse Person erstellen
anna = Person("Anna", 16)
jan = Person("Jan", 17)

# Methoden aufrufen
anna.vorstellen()  # Ausgabe: Hallo, ich bin Anna und 16 Jahre alt.
jan.vorstellen()   # Ausgabe: Hallo, ich bin Jan und 17 Jahre alt.

# Geburtstag feiern
anna.geburtstag_feiern()  # Ausgabe: Anna ist jetzt 17 Jahre alt.
```



`class` ... **ist der Klassenname,** `self` **bezieht sich auf die aktuelle Instanz.**

Beispiel: Personen

```python
class Person:
    def __init__(self, name, alter):
        self.name = name
        self.alter = alter

    def vorstellen(self):
        print(f"Hallo, ich bin {self.name} und {self.alter} Jahre alt.")

    def geburtstag_feiern(self):
        self.alter += 1
        print(f"{self.name} ist jetzt {self.alter} Jahre alt.")



# Objekte der Klasse Person erstellen
anna = Person("Anna", 16)
jan = Person("Jan", 17)

# Methoden aufrufen
anna.vorstellen()  # Ausgabe: Hallo, ich bin Anna und 16 Jahre alt.
jan.vorstellen()   # Ausgabe: Hallo, ich bin Jan und 17 Jahre alt.

# Geburtstag feiern
anna.geburtstag_feiern()  # Ausgabe: Anna ist jetzt 17 Jahre alt.
```



`class` ... ist der Klassenname, `self` bezieht sich auf die aktuelle Instanz.

Beispiel: Personen

```python
class Person:
    def __init__(self, name, alter):
        self.name = name
        self.alter = alter

    def vorstellen(self):
        print(f"Hallo, ich bin {self.name} und {self.alter} Jahre alt.")

    def geburtstag_feiern(self):
        self.alter += 1
        print(f"{self.name} ist jetzt {self.alter} Jahre alt.")



# Objekte der Klasse Person erstellen
anna = Person("Anna", 16)
jan = Person("Jan", 17)

# Methoden aufrufen
anna.vorstellen()  # Ausgabe: Hallo, ich bin Anna und 16 Jahre alt.
jan.vorstellen()   # Ausgabe: Hallo, ich bin Jan und 17 Jahre alt.

# Geburtstag feiern
anna.geburtstag_feiern()  # Ausgabe: Anna ist jetzt 17 Jahre alt.
```

Instanzen erstellen (initialisieren)

Methode

Methode

`class` ... **ist der Klassenname,** `self` **bezieht sich auf die aktuelle Instanz.**

Aufgaben

Programmier-Skript

- Lesen Sie Definition 7.1, Beispiele 7.1-7.2

  → Ausführen und zwei weitere Instanzen von `Person` erstellen.
- Aufgaben 7.1, 7.2 → Zuerst auf VS Code, dann Test auf Moodle
- Schnelle: Aufgabe 7.3, 7.4 (Skript selber lesen)

Zusammenfassung: Klassen

- Klasse = eine Art von Variable

Zusammenfassung: Klassen

- Klasse = eine Art von Variable
- **Variable** = eine Instanz der Klasse

Zusammenfassung: Klassen

- Klasse = eine Art von Variable
- **Variable** = eine Instanz der Klasse
- Z.B. `meineliste=[10,6,1]` $\Rightarrow$ `list` (Instanz der Klasse `list`)

Zusammenfassung: Klassen

- Klasse = eine Art von Variable
- **Variable** = eine Instanz der Klasse
- Z.B. `meineliste=[10,6,1]` $\Rightarrow$ `list` (Instanz der Klasse `list`)
- Klassen können **Attribute** und **Methoden** haben: z.B. `hund = Hund("Bello", 3)` $\Rightarrow$ `hund.name` (Attribut) und `hund.bellen()` (Methode)