

# DATA SCIENCE 1

TUTORIAL DAY 2

PROF. DR. CHRISTIAN BOCKERMANN

HOCHSCHULE BOCHUM

WINTERSEMESTER 2021/2022

1 Wiederholung Python Basics

2 Eigene Funktionen (Beispiel)

3 Von Excel (VBA) zu Pandas

## Python

- Skript-Sprache, Text-Datei wird ausgeführt

## Einfach Datentypen

- Zahlen als **int** und **float**
- Texte als **str**

## Python

- Skript-Sprache, Text-Datei wird ausgeführt

## Einfach Datentypen

- Zahlen als **int** und **float**
- Texte als **str**

## Mengen von Daten

- Listen mit **list** und **[]**
- Mengen mit **set**

## Python Code-Blöcke

- Code-Blöcke durch Einrückung (Leerzeichen **oder** Tab)

```
if zahl > 50:  
    steuersatz = 120  
    sum = sum + zahl  
  
preis = sum * 2
```

## Python Code-Blöcke

- Code-Blöcke durch Einrückung (Leerzeichen **oder** Tab)

```
if zahl > 50:  
    steuersatz = 120  
    sum = sum + zahl  
preis = sum * 2
```

## Python Code-Blöcke

- Code-Blöcke durch Einrückung (Leerzeichen **oder** Tab)

```
if zahl > 50:  
    steuersatz = 120  
    sum = sum + zahl  
preis = sum * 2
```

## Python Code-Blöcke

```
zahlen = [1,2,3,4,5,6]
gerade = []
ungerade = []

for zahl in zahlen:
    if zahl % 2 == 0:
        gerade.append(zahl)
    else:
        ungerae.append(zahl)

print("Die geraden Zahlen:")
print(gerade)
```

## Python Code-Blöcke

```
zahlen = [1,2,3,4,5,6]
gerade = []
ungerade = []

for zahl in zahlen:
    if zahl % 2 == 0:
        gerade.append(zahl)
    else:
        ungerae.append(zahl)

print("Die geraden Zahlen:")
print(gerade)
```

## Python Code-Blöcke

```
zahlen = [1,2,3,4,5,6]
gerade = []
ungerade = []

for zahl in zahlen:
    if zahl % 2 == 0:
        gerade.append(zahl)
    else:
        ungerade.append(zahl)

print("Die geraden Zahlen:")
print(gerade)
```

## Python Code-Blöcke

```
zahlen = [1,2,3,4,5,6]
gerade = []
ungerade = []

for zahl in zahlen:
    if zahl % 2 == 0:
        gerade.append(zahl)
    else:
        ungerae.append(zahl)

print("Die geraden Zahlen:")
print(gerade)
```

## Python Code-Blöcke

```
zahlen = [1,2,3,4,5,6]
gerade = []
ungerade = []

for zahl in zahlen:
    if zahl % 2 == 0:
        gerade.append(zahl)
    else:
        ungerade.append(zahl)

print("Die geraden Zahlen:")
print(gerade)
```

## if-Bedingung

```
zahl = 32

if zahl > 50:
    text = "viel"
else:
    text = "wenig"

print("Ganz schoen " + text + ".")
```

## if-Bedingung

```
zahl = 32
text = "wenig"

if zahl > 50:
    text = "viel"

print("Ganz schoen " + text + ".")
```

Variante ohne **else**

## for-Schleife

```
zahlen = [0, 1, 1, 2, 3, 5, 8]
sum = 0
anzahl = 0

for zahl in zahlen:
    sum = sum + zahl
    anzahl = anzahl + 1

schnitt = sum / anzahl
```

## for-Schleife

```
zahlen = [0, 1, 1, 2, 3, 5, 8]
sum = 0
anzahl = 0

for zahl in zahlen:
    sum = sum + zahl
    anzahl = anzahl + 1

schnitt = sum / anzahl
```

Schleifen-Block durch Einrückung!

# Eigene Funktionen (Beispiel)

## Eigene Funktionen mit **def**

```
def durchschnitt(zahlen):  
    sum = 0  
    for zahl in zahlen:  
        sum = sum + zahl  
  
    return sum / len(zahlen)
```

## Eigene Funktionen mit **def**

```
def durchschnitt(zahlen):  
    sum = 0  
    for zahl in zahlen:  
        sum = sum + zahl  
  
    return sum / len(zahlen)
```

- **return** legt Funktionsergebnis fest
- Bei **return** endet die Funktion sofort

## Eigene Funktionen mit **def**

```
def durchschnitt(zahlen):  
    sum = 0  
    if len(zahlen) == 0:  
        return 0  
  
    for zahl in zahlen:  
        sum = sum + zahl  
  
    return sum / len(zahlen)
```

- **return** legt Funktionsergebnis fest
- Bei **return** endet die Funktion sofort

## Wirtschaftsinformatik 1, Aufgabenblatt 4

### Aufgabenblatt 4: Klausurbeispiele für benutzerdefinierte Funktionen

#### Aufgabe 4\_1

Sie arbeiten beim Steueramt der Stadt Bochum und sollen ein VBA-Programm schreiben, mit dem die Hundesteuer berechnet werden kann.

Bei der Berechnung der Steuern wird zwischen Normalhunden und Kampfhunden unterschieden.

Für Normalhunde wird ein Staffelsteuersatz verwendet:

- Bei einem Hund im Haushalt kostet der Hund 120 € pro Jahr
- Bei zwei oder drei Hunden im Haushalt kostet jeder Hund 150 € pro Jahr
- Bei vier und mehr Hunden im Haushalt kostet jeder Hund 180 € pro Jahr

Kampfhunde werden grundsätzlich ebenfalls wie Normalhunde behandelt (das heißt die Anzahl von Kampfhunden und Normalhunden wird addiert und danach nach obiger Staffelung der Steuersatz berechnet), es gilt jedoch folgende Sonderregelung:

- Übersteigt die Anzahl der Kampfhunde die Anzahl der Normalhunde, so wird der zu entrichtende Steuersatz verdoppelt.

Die Steuersätze werden auf Konstanten abgelegt. Unten dargestellte Benutzer-Schnittstelle soll realisiert werden:

D2		fx =BERECHNEHUNDESTEUER(B2;C2)			
	A	B	C	D	E
1	Steuernummer	Normalhunde	Kampfhunde	Preis	
2	1234	3	5	2880	
3	1235	4	0	720	

## Wie lösen Informatiker Probleme?

## Wie lösen Informatiker Probleme?

1. In kleine Probleme zerteilen
2. Kleine Probleme lösen
3. Lösungen zusammensetzen

## Wie lösen Informatiker Probleme?

1. In kleine Probleme zerteilen
2. Kleine Probleme lösen
3. Lösungen zusammensetzen

**Funktionen eignen sich gut, um Teilprobleme zu lösen!**

## Hundesteuern, Wirtschaftsinformatik 1, Aufgabenblatt 4

- Wie werden die Steuern grundsätzlich berechnet?
- Welche Eingabewerte braucht man?
- Welche Ausnahmen gibt es? Was ändert sich dann?
- Weitere Sonderregeln?

## Hundesteuern, Wirtschaftsinformatik 1, Aufgabenblatt 4

- Wie werden die Steuern grundsätzlich berechnet?
- Welche Eingabewerte braucht man?
- Welche Ausnahmen gibt es? Was ändert sich dann?
- Weitere Sonderregeln?

### **Aufgabe:**

- Schreiben Sie die Funktion `hundesteuer( . . )`

## Demo:

### Entwicklung einer Lösung für die Hundesteuer-Aufgabe

- Wieso sind Teilprobleme gut?
- Wie finde ich ggf. Programmierfehler?

# Von Excel (VBA) zu Pandas

## Daten zur Aufgabe

Excel-Datei unter

```
Kurse/DataScience1/data/hundesteuer.xls
```

In Python mit Pandas:

```
import pandas as pd  
  
df = pd.read_excel("Kurse/../hundesteuer.xls")
```

Steuernummer	Normalhunde	Kampfhunde	Preis
<b>1234</b>	<b>3</b>	<b>5</b>	<b>2880</b>
<b>1235</b>	<b>4</b>	<b>0</b>	<b>720</b>
<b>1236</b>	<b>1</b>	<b>0</b>	<b>120</b>

	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>
	Steuernummer	Normalhunde	Kampfhunde	Preis
<b>0</b>	1234	3	5	2880
<b>1</b>	1235	4	0	720
<b>2</b>	1236	1	0	120

	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>
	Steuernummer	Normalhunde	Kampfhunde	Preis
<b>0</b>	1234	3	5	2880
<b>1</b>	1235	4	0	720
<b>2</b>	1236	1	0	120

```
df.iloc[ .. ]
```

	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>
	Steuernummer	Normalhunde	Kampfhunde	Preis
<b>0</b>	1234	3	5	2880
<b>1</b>	1235	4	0	720
<b>2</b>	1236	1	0	120

```
df.iloc[ 0, 2 ]
```

## Zeilenweiser Zugriff

```
df = pd.read_excel("../hundesteuer.xls")

for i in range(len(df)):
    normalhunde = df.iloc[i, 1]
    kampfunde = df.iloc[i, 2]

    print("zeile: " + i)
    print("  normal: " + normalhunde)
    print("  kampf:  " + kampfunde)
```

## Zeile in DataFrame = Series

```
zeile = df.iloc[0,:]  
  
#zeile ist dann ein Series Objekt:  
#  
# Steuernummer      1234  
# Normalhunde       3  
# Kampfhunde        5  
# Preis              2880  
# Steuern            42  
#Name: A, dtype: int64
```

## Zeile in DataFrame

```
zeile = df.iloc[0,:]  
  
normal = zeile['Normalhunde']  
kampf = zeile['Kampfhunde']
```

## Zeile in DataFrame

```
zeile = df.iloc[0,:]  
  
normal = zeile['Normalhunde']  
kampf = zeile['Kampfhunde']
```

## Hundesteuern aus Zeile (Series) berechnen:

```
def berechneHundesteuer(zeile):  
    normal = zeile['Normalhunde']  
    kampf = zeile['Kampfhunde']  
    return hundesteuer(normal, kampf)
```

## Funktion für jede Zeile aufrufen: `df.apply`

```
# rufe berechneHundesteuer fuer jede Zeile auf  
df.apply(berechneHundesteuer, axis=1)
```

## Funktion für jede Zeile aufrufen: `df.apply`

```
# rufe berechneHundesteuer fuer jede Zeile auf  
df.apply(berechneHundesteuer, axis=1)
```

- Ergebnis ist Series Objekt
- Series enthält Ergebnis der Funktion für jede Zeile