

# Aufgaben.

## Aufgabe 1 | Hello World

Schreibe ein Programm das zunächst den Text „Das ist mein zweites Programm.“ ausgibt. Füge danach einen weiteren print()-Befehl ein, der diesen Text anzeigt: „Dieses gibt nur einen Text aus.“

## Aufgabe 2 | Variablen und Operatoren

- a) Schreibe ein Programm, das zwei Variablen enthält. Weise diesen jeweils eine Zahl zu. Erstelle danach eine Ausgabe, die zunächst die Werte dieser beiden Variablen ausgibt und danach das Ergebnis, wenn man diese addiert.
- b) Frage den Anwender nach seinem Alter. Erstelle danach eine Ausgabe, die anzeigt, wie alt er in 5 Jahren ist.

## Aufgabe 3 | Strings

- a) Speichere das Wort „Python“, ein Leerzeichen, einen Bindestrich und das Wort „Kurs“ jeweils in einer eigenen Variablen.
  - 1) Gib den Begriff „Python - Kurs“ aus. Verwende dazu ausschließlich die soeben erstellten Variablen.
  - 2) Gib drei Mal hintereinander Python aus - jeweils mit einem Leerzeichen dazwischen. Der print()-Befehl soll die Variablen mit dem Wort „Python“ und mit dem Leerzeichen jeweils nur einmal enthalten.  
(Tipp: Hierfür musst du zunächst den Begriff „Python“ mit dem Leerzeichen zusammenfügen. Wenn du diesen Ausdruck in eine Klammer stellst, kannst du ihn als Ganzes multiplizieren)
  - 3) Gib die ersten drei Buchstaben des Begriffs „Python“ in einem print()-Befehl aus. Verwende daraufhin einen neuen print()-Befehl für die zweite Worthälfte.
- b) Speichere folgenden Text in einer Variable ab: „Hier steht ein beliebiger Text.“
  - 1) Erstelle einen print()-Befehl, der anzeigt, wie oft der Buchstabe ‚e‘ in diesem Text enthalten ist.
  - 2) Gib zunächst das erste Wort dieses Texts in Großbuchstaben aus. Im übrigen Text soll der Buchstabe ‚i‘ durch die Zahl 1 ersetzt werden. Verwende hierfür nur einen print()-Befehl.

## Aufgabe 4 | Listen und Listenfunktionen

- a) Erstelle eine Liste, die die Namen der Städte Berlin, Hamburg, Köln und Stuttgart enthält.
- 1) Gib das erste und das letzte Element der Liste aus. Wähle den Index so, dass die print()-Funktion immer das letzte Element ausgibt - unabhängig von der Länge der Liste.
  - 2) Füge die Stadt Frankfurt zur Liste hinzu und gib die gesamte Liste aus.
  - 3) Sortiere die Liste und gib sie komplett aus.
- b) Speichere in einer Liste die Daten von drei Freunden. Diese sollen jeweils aus einer weiteren Liste bestehen, die den Vornamen und das Alter enthalten. Gib die komplette Liste aus.

## Aufgabe 5 | Tupel

- a) Erstelle ein Tupel, das den Vornamen, den Nachnamen und das Alter einer Person enthält. Füge - nachdem du das Tupel bereits erstellt hast - auch noch die Telefonnummer hinzu.
- b) Die im Tupel festgehaltene Person hatte Geburtstag. Daher willst du das Alter ändern. Da die Daten bei Tupeln jedoch unveränderlich sind, ist das beim Programm aus Teilaufgabe a) nicht möglich. Erstelle daher in einem neuen Programm das Tupel so, dass eine Änderung des Alters möglich ist. Erhöhe daraufhin den Wert um 1 und gib das Tupel aus.

## Aufgabe 6 | Dictionaries

Wenn du dir nochmals das Tupel aus Aufgabe 5 a) anschaust, stellst du fest, dass nicht sofort klar wird, worauf sich die einzelnen Angaben beziehen. Verwende deshalb ein Dictionary, das passende Bezeichner für den Vor- und den Nachnamen sowie das Alter verwendet. Füge in das fertige Dictionary anschließend noch einen Eintrag für die Telefonnummer ein. Gib danach den Vornamen und die Telefonnummer aus.

## Aufgabe 7 | Konvertierungen zwischen Datentypen

Erstelle ein Programm und fordere den Leser dazu auf, eine Zahl einzugeben.

- 1) Speichere die Eingabe in einer Variable ab, ohne deren Typ zu verändern.
- 2) Erstelle einen print-Befehl und addiere darin diese Variable mit sich selbst.
- 3) Gib den Datentyp der Variable aus.
- 4) Wandle den Wert in einen int-Wert um und speichere ihn in einer neuen Variable ab.
- 5) Gib in einem weiteren print-Befehl die Addition der neuen Variable aus.

## Aufgabe 8 | if, elif, else

- a) Schreibe ein Programm für ein Geschäft und definiere darin zwei boolesche Variablen: sonntag und feiertag. Erstelle eine Kontrollstruktur, die:
- 1) an allen Sonntagen (auch wenn es sich dabei um einen Feiertag handelt) den Leser darauf hinweist, dass das Geschäft geschlossen ist und ihm einen schönen Sonntag wünscht.
  - 2) an Feiertagen, die nicht auf einen Sonntag fallen, den Leser lediglich darauf hinweist, dass das Geschäft geschlossen ist.
  - 3) an allen übrigen Tagen mitteilt, dass das Geschäft geöffnet ist.
- b) Probiere das Programm mit allen möglichen Kombinationen für die beiden Variablen aus.

## Aufgabe 9 | Vergleichsoperatoren

Schreibe ein Programm für eine Bank. Eine Variable soll dabei den Kontostand eines Kunden enthalten. Frage den Kunden daraufhin, wie viel Geld er abheben will.

- 1) Passe den Kontostand an, wenn der Betrag verfügbar ist und gib das neue Guthaben aus.
- 2) Teile dem Kunden mit, dass sein Konto leer ist, wenn der Abhebebetrag dem Kontostand entspricht.
- 3) Gib eine Fehlermeldung aus, wenn der Abhebebetrag größer als der Kontostand ist. Gib das Guthaben in diesem Fall unverändert aus.

## Aufgabe 10 | Der in-Operator

Erstelle eine Liste mit den Namen von fünf Freunden. Überprüfe daraufhin mit dem in-Operator, ob sich der Name Michael in der Liste befindet.

## Aufgabe 11 | Logische Operatoren – and, or, not

Schreibe ein Programm, das die Aufträge in einem Warenlager bearbeitet.

- a) Das erste Programm soll zum einen den Warenbestand für ein bestimmtes Produkt enthalten. Zum anderen ist eine boolesche Variable vorhanden, die angibt, ob im Moment ein Mitarbeiter verfügbar ist. Frage vom Anwender die gewünschte Menge ab. Falls diese vorhanden und außerdem ein Mitarbeiter verfügbar ist, soll das Programm eine entsprechende Nachricht ausgeben. Ist dies nicht der Fall, soll es anzeigen, dass die Bearbeitung im Moment nicht möglich ist.
- b) Das Unternehmen unterhält noch einen weiteren Standort, an dem das entsprechende Produkt ebenfalls vorrätig ist. Schreibe ein neues Programm, das die Bestände an beiden Standorten enthält. Es soll dem Anwender ebenfalls die Eingabe der gewünschten Anzahl ermöglichen. Es soll dann lediglich überprüfen, ob diese an einem der beiden Standorte verfügbar ist.

## Aufgabe 12 | Schleifen

- a) Erstelle eine while-Schleife. Diese soll eine Zahl vom Anwender abfragen und den entsprechenden Wert in eine Liste einfügen. Danach soll sie abfragen, ob er mit der Eingabe fortfahren will. Gib anschließend die Liste aus.
- b) Erstelle eine for-Schleife, die die Quadratzahlen von 1 bis 20 ausgibt.

## Aufgabe 13 | Stringformatierung

Erstelle eine Variable mit einer Zeichenkette und eine weitere Variable mit einer Zahl. Gib die Werte der beiden Variablen in einem einzigen print-Befehl zusammen mit einem erklärenden Begleittext aus.

## Aufgabe 14 | Eingabe (Dateien lesen)

Erstelle eine Textdatei mit den Namen von fünf Freunden. Schreibe daraufhin ein Programm, das die Namen einliest. Daraufhin soll das Programm sie einzeln mit einer Begrüßungsformel versehen und jeweils in einer eigenen Zeile ausgeben. Verwende hierfür eine for-Schleife.

## Aufgabe 15 | Ausgabe (Dateien schreiben)

Verwende wieder die gleiche Textdatei wie in Aufgabe 14. Schreibe nun ein Programm, das es dem Anwender ermöglicht, neue Freunde einzutragen. Die bisherigen Einträge sollen dabei erhalten bleiben. Die Abfrage der Namen soll in einer Schleife erfolgen. Frage den Anwender nach jedem Durchgang, ob er einen weiteren Namen hinzufügen will.

## Aufgabe 16 | set und frozenset

Erstelle ein Set, das die Namen von drei Freunden enthält.

- 1) Füge einen weiteren Namen in das Set ein.
- 2) Erstelle daraufhin eine weitere Liste, die die Freunde deiner Freundin bzw. deines Friends enthält.
- 3) Nun wollt ihr ein gemeinsames Fest planen und dazu nur die Freunde einladen, die mit beiden befreundet sind. Gestalte eine entsprechende Ausgabe.

## Aufgabe 17 | Sortierung (sort, sorted)

- Erstelle eine Liste mit den Namen deiner Freunde. Sortiere sie in umgekehrter Reihenfolge. Die sortierte Liste soll unter der gleichen Variablen abgespeichert werden wie die ursprüngliche Version.
- Schreibe ein Programm, das den Anwender zur Eingabe fünf beliebiger Zahlen auffordert und diese in einer Liste speichert. Im weiteren Verlauf soll diese Liste erhalten bleiben, um die Reihenfolge der Eingaben nachvollziehen zu können. Allerdings ist auch eine sortierte Version erforderlich. Gestalte das Programm so, dass beide Listen verfügbar sind und gib sie anschließend aus.

## Aufgabe 18 | Iteratoren

Erstelle ein Set mit vier Namen. Gestalte daraufhin einen Iterator und gib die Inhalte einzeln aus.

## Aufgabe 19 | Copy und Deep Copy

- Erstelle eine Liste mit vier Städten. Erstelle daraufhin eine Kopie dieser Liste und ändere darin einen Eintrag. Gib daraufhin beide Listen aus.
- Erstelle nun eine Liste, die vier Städte mit ihrer jeweiligen Einwohnerzahl enthält. Kopiere diese Liste, ändere einen Eintrag ab und gib beide Listen aus.

## Aufgabe 20 | Funktionen selbst schreiben

Schreibe eine Funktion, die den Wert einer beliebigen quadratischen Gleichung an einem bestimmten Punkt berechnet. Quadratische Gleichungen sind nach folgendem Schema aufgebaut:  $ax^2 + bx + c$ . Verwende die Werte  $a$ ,  $b$ ,  $c$  und  $x$  als Übergabewerte und gib das Ergebnis direkt in der Funktion aus. Das Hauptprogramm soll nur die Funktion aufrufen.

## Aufgabe 21 | Globale und lokale Variablen

Ändere das Programm aus Aufgabe 20 so ab, dass es keine Übergabewerte, sondern globale Variablen verwendet, um die Werte zu übermitteln. Die Funktionsweise soll dabei erhalten bleiben.

## Aufgabe 22 | Positionsparameter und Schlüsselwortparameter

Betrachte nochmals das Programm aus Aufgabe 20. Ändere dieses nun so ab, dass es die Funktion auch ausführen kann, wenn der Anwender die Funktion nur mit den Werten a, b und c aufruft. In diesem Fall soll die Funktion für den Wert x automatisch die Zahl 0 verwenden. Rufe die Funktion im Hauptprogramm einmal wie bisher mit vier Parametern auf und daraufhin nur mit drei Übergabewerten.

## Aufgabe 23 | Rückgabewerte (return)

Erstelle eine weitere Version der Funktion aus Aufgabe 20. Diese soll nun das Ergebnis nicht direkt ausgeben, sondern als Rückgabewert an das Hauptprogramm zurückgeben. Rufe dort die Funktion auf, speichere das Ergebnis in einer Variable ab und gib diese aus.

## Aufgabe 24 | Eingebaute Funktionen – Standardbibliothek

Schreibe ein Programm, das eine beliebige Zahl vom Anwender abfragt. Gib daraufhin den Betrag dieses Wertes aus. Suche dafür eine passende Funktion in der Standardbibliothek.

## Aufgabe 25 | Objektorientierte Programmierung

Stell dir vor, du erstellst ein Programm für ein Restaurant. Dieses soll die Gäste registrieren, ihre Tischnummer und die bestellten Speisen festhalten und den Rechnungsbetrag berechnen.

- 1) Erstelle eine Klasse für einen Besucher. Diese soll als Attribute die Tischnummer, eine Liste für die bestellten Gerichte und Getränke (die zunächst leer ist) und den Rechnungsbetrag (der zu Beginn bei 0 Euro liegt) enthalten. Verwende für den Konstruktor daher nur die Tischnummer als Übergabewert und gib die übrigen Attribute direkt vor.
- 2) Erstelle im Hauptprogramm ein Dictionary für die Speisekarte. Der Schlüssel soll dabei der Name der Speise beziehungsweise des Getränkes sein. Gib als Wert den entsprechenden Preis an.
- 3) Erstelle eine Methode, die es erlaubt, ein Gericht oder ein Getränk zu bestellen. Diese soll als Übergabewert den Namen des bestellten Gerichtes/Getränktes erhalten (also den Schlüssel für das Dictionary). Füge diesen in die Liste ein und addiere den zugehörigen Preis zum bisherigen Rechnungsbetrag.
- 4) Erstelle eine Methode, die die alle Daten des Gastes ausgibt - also die Tischnummer, die bestellten Speisen und den Rechnungsbetrag. Die bestellten Gerichte sollen dabei untereinander jeweils in einer einzelnen Zeile ausgegeben werden.
- 5) Erstelle im Hauptprogramm ein Objekt für einen neuen Gast. Dieser bestellt zwei Gerichte und zwei Getränke.
- 6) Gib daraufhin die Daten des Gastes aus.

# Lösungen.

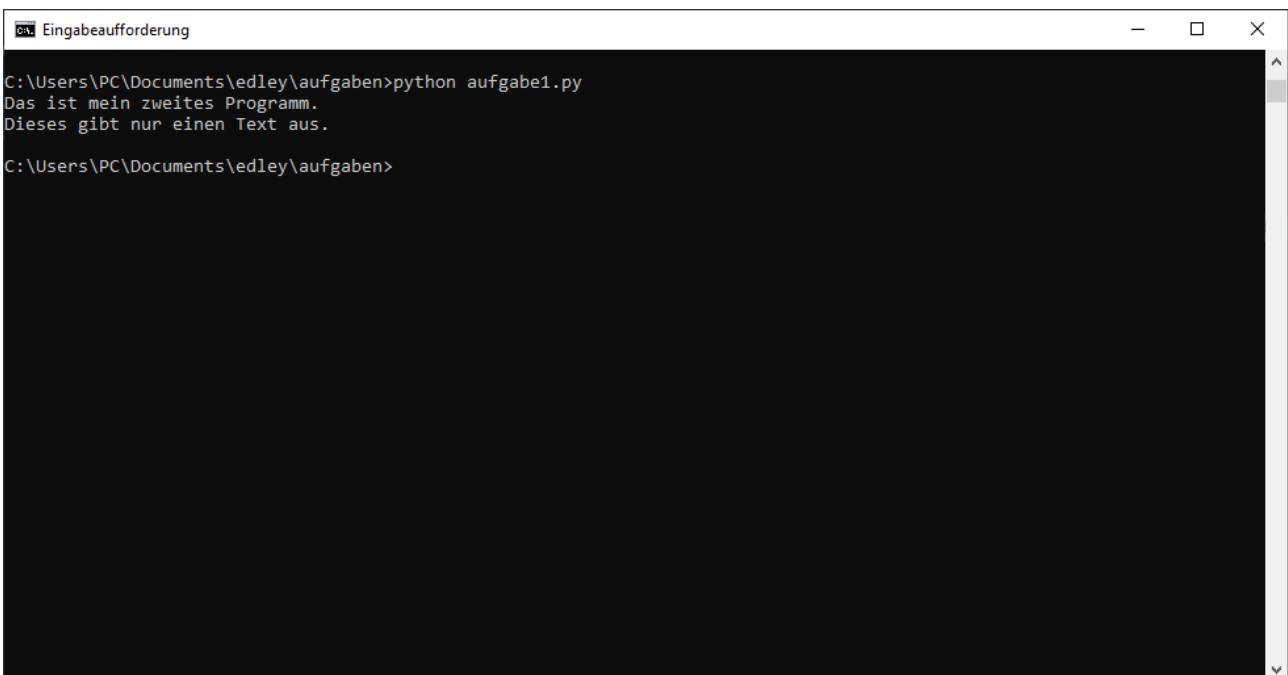
---

**Hinweis 1:** Es gibt viele verschiedene Möglichkeiten, die Aufgaben richtig zu lösen. Die folgenden Lösungen sind daher nur als mögliche Lösungsvorschläge zu verstehen. Diese können von deiner Version abweichen. Wichtig ist, dass das Ergebnis in der Ausgabe stimmt und dein Programm funktioniert.

**Hinweis 2:** Du siehst in den Screenshots die Eingabeaufforderung - es funktioniert aber natürlich auch in der Konsolenausgabe von PyCharm oder jeder anderen IDE.

## Lösung 1 | Hello World

```
print("Das ist mein zweites Programm.")  
print("Dieses gibt nur einen Text aus.")
```



```
Eingabeaufforderung  
C:\Users\PC\Documents\edley\aufgaben>python aufgabe1.py  
Das ist mein zweites Programm.  
Dieses gibt nur einen Text aus.  
C:\Users\PC\Documents\edley\aufgaben>
```

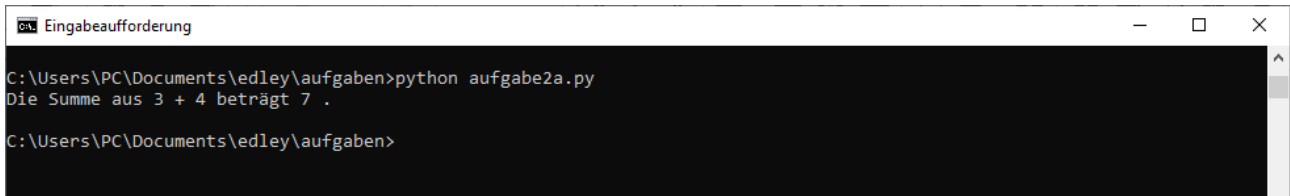
## Lösung 2 | Variablen und Operatoren

a)

```
a = 3
```

```
b = 4
```

```
print("Die Summe aus", a, "+", b, "beträgt", a + b)
```

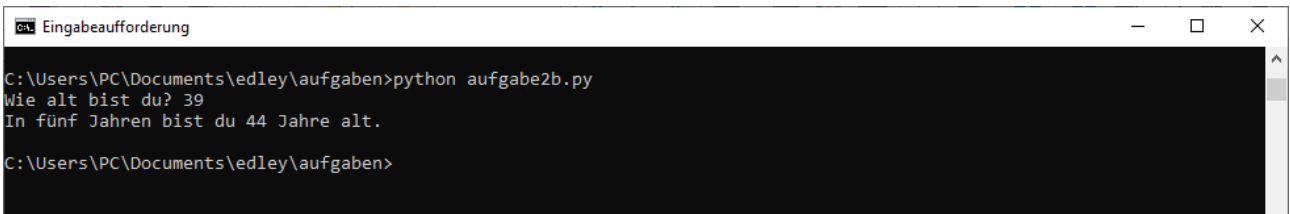


```
ca\ Eingabeaufforderung
C:\Users\PC\Documents\edley\aufgaben>python aufgabe2a.py
Die Summe aus 3 + 4 beträgt 7 .
C:\Users\PC\Documents\edley\aufgaben>
```

b)

```
alter = int(input("Wie alt bist du? "))
```

```
print("In fünf Jahren bist du", alter + 5, "Jahre alt.")
```



```
ca\ Eingabeaufforderung
C:\Users\PC\Documents\edley\aufgaben>python aufgabe2b.py
Wie alt bist du? 39
In fünf Jahren bist du 44 Jahre alt.
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 3 | Strings

a)

```
variable1 = "Python"
```

```
variable2 = " "
```

```
variable3 = "-"
```

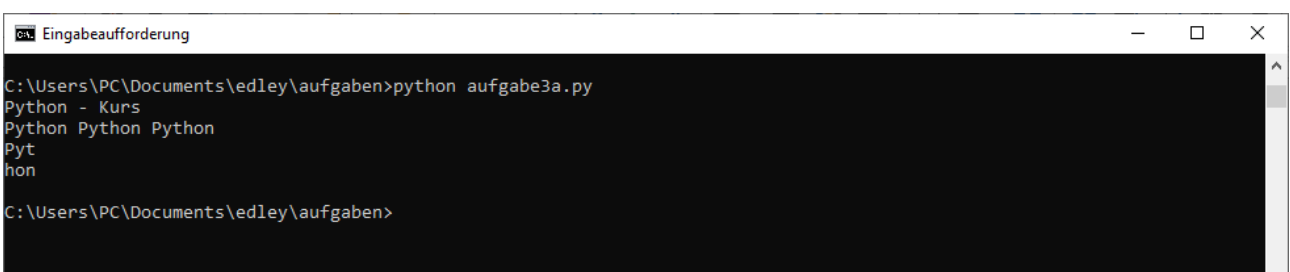
```
variable4 = "Kurs"
```

```
print(variable1 + variable2 + variable3 + variable2 + variable4)
```

```
print((variable1 + variable2)*3)
```

```
print(variable1[:3])
```

```
print(variable1[3:])
```

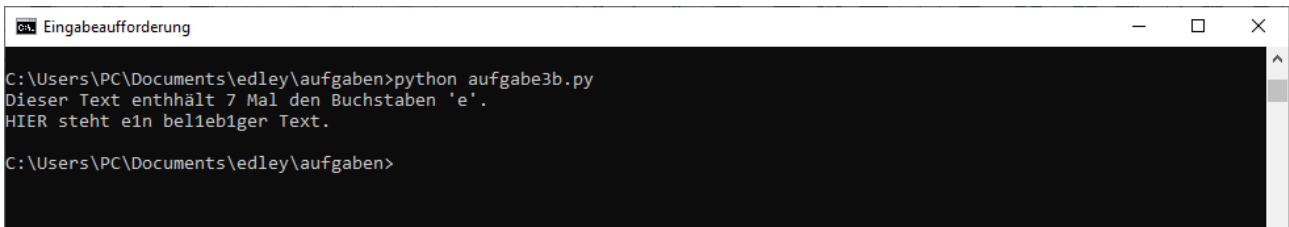


```
ca\ Eingabeaufforderung
C:\Users\PC\Documents\edley\aufgaben>python aufgabe3a.py
Python - Kurs
Python Python Python
Pyt
hon
C:\Users\PC\Documents\edley\aufgaben>
```



b)

```
text = "Hier steht ein beliebiger Text."  
print("Dieser Text enthhält", text.count("e"), "Mal den Buchstaben 'e'.")  
print(text[:4].upper() + text[4:].replace("i", "1"))
```



```
ca. Eingabeaufforderung  
C:\Users\PC\Documents\edley\aufgaben>python aufgabe3b.py  
Dieser Text enthhält 7 Mal den Buchstaben 'e'.  
HIER steht ein beliebiger Text.  
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 4 | Listen und Listenfunktionen

a)

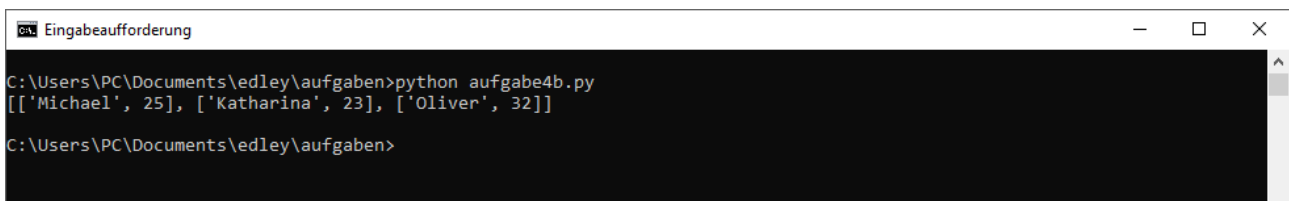
```
liste = ["Berlin", "Hamburg", "Köln", "Stuttgart"]  
print(liste[0])  
print(liste[-1])  
liste.append("Frankfurt")  
print(liste)  
liste.sort()  
print(liste)
```



```
ca. Eingabeaufforderung  
C:\Users\PC>cd documents\edley\aufgaben  
C:\Users\PC\Documents\edley\aufgaben>python aufgabe4a.py  
Berlin  
Stuttgart  
['Berlin', 'Hamburg', 'Köln', 'Stuttgart', 'Frankfurt']  
['Berlin', 'Frankfurt', 'Hamburg', 'Köln', 'Stuttgart']  
C:\Users\PC\Documents\edley\aufgaben>
```

b)

```
liste = [["Michael", 25], ["Katharina", 23], ["Oliver", 32]]  
print(liste)
```

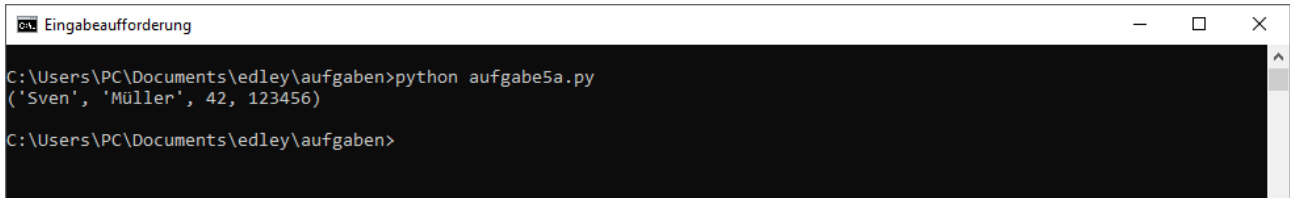


```
ca. Eingabeaufforderung  
C:\Users\PC\Documents\edley\aufgaben>python aufgabe4b.py  
[['Michael', 25], ['Katharina', 23], ['Oliver', 32]]  
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 5 | Tupel

a)

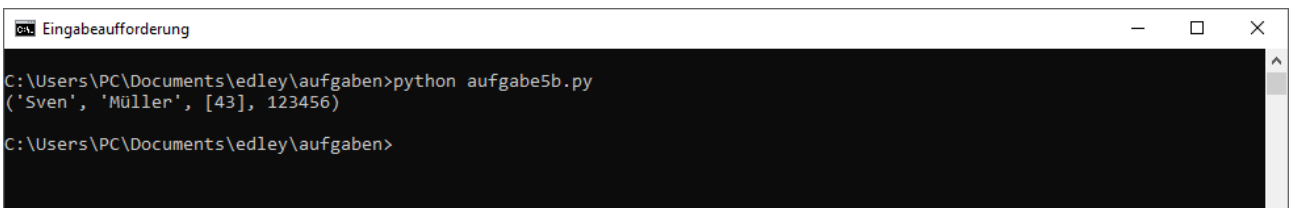
```
tupel = "Sven", "Müller", 42
tupel += 123456,
print(tupel)
```



```
Eingabeaufforderung
C:\Users\PC\Documents\edley\aufgaben>python aufgabe5a.py
('Sven', 'Müller', 42, 123456)
C:\Users\PC\Documents\edley\aufgaben>
```

b)

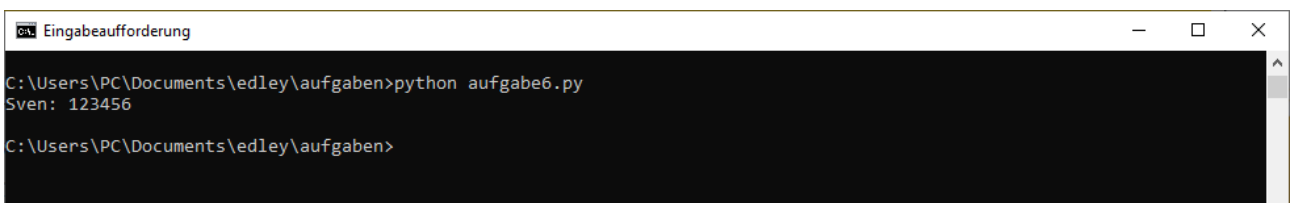
```
tupel = "Sven", "Müller", [42], 123456
tupel[2][0] += 1
print(tupel)
```



```
Eingabeaufforderung
C:\Users\PC\Documents\edley\aufgaben>python aufgabe5b.py
('Sven', 'Müller', [43], 123456)
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 6 | Dictionaries

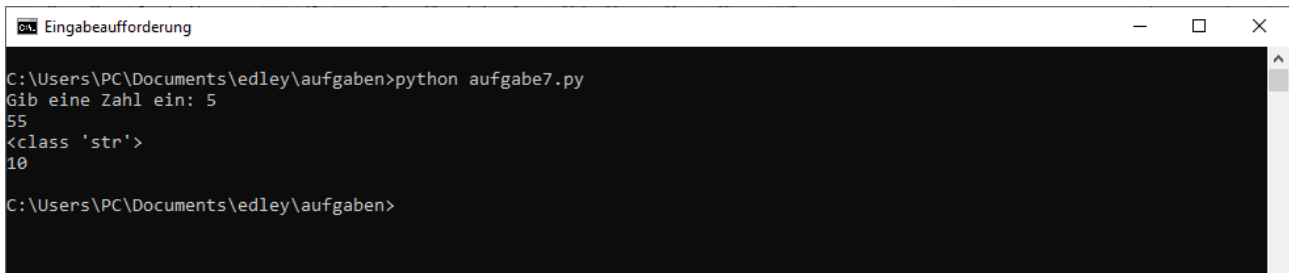
```
dictionary = {"vorname": "Sven", "nachname": "Müller", "alter": 42}
dictionary["telefonnummer"] = 123456
print(dictionary["vorname"] + ":", dictionary["telefonnummer"])
```



```
Eingabeaufforderung
C:\Users\PC\Documents\edley\aufgaben>python aufgabe6.py
Sven: 123456
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 7 | Konvertierungen zwischen Datentypen

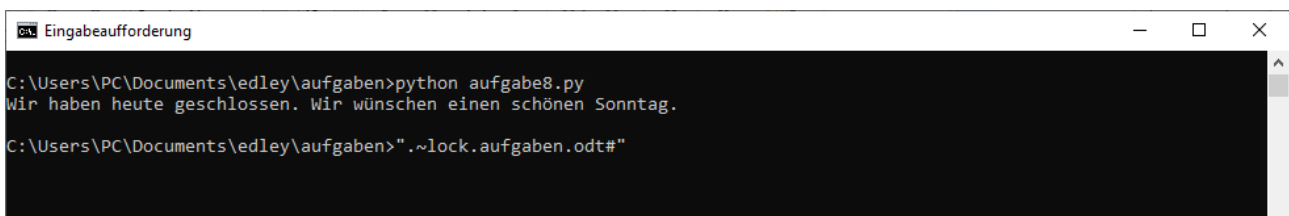
```
eingabe = input("Gib eine Zahl ein: ")
print(eingabe + eingabe)
print(type(eingabe))
zahl = int(eingabe)
print(zahl + zahl)
```



```
C:\Users\PC\Documents\edley\aufgaben>python aufgabe7.py
Gib eine Zahl ein: 5
55
<class 'str'>
10
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 8 | if, elif, else

```
sonntag = True
feiertag = False
if sonntag:
    print("Wir haben heute geschlossen. Wir wünschen einen schönen Sonntag.")
elif feiertag:
    print("Heute haben wir nicht geöffnet.")
else:
    print("Herzlich willkommen! Wir haben geöffnet")
```



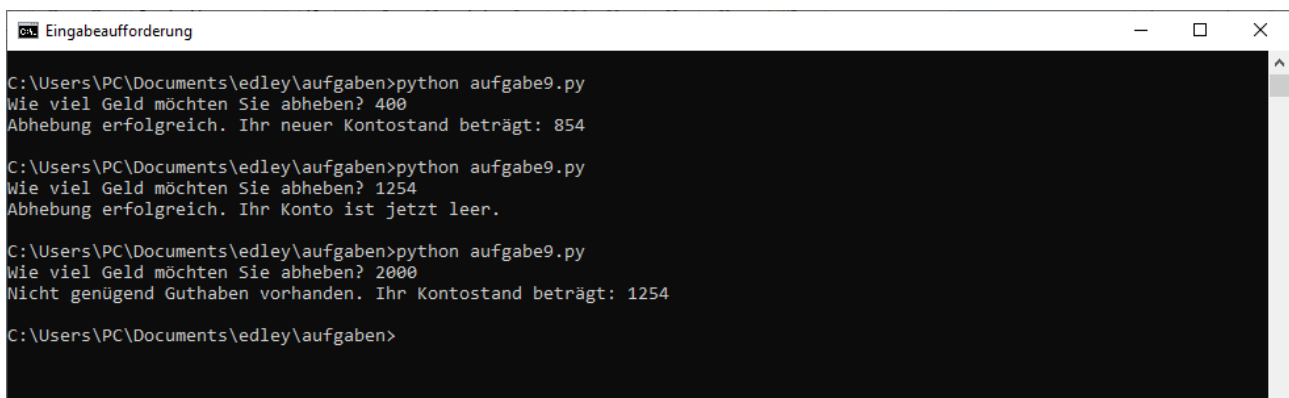
```
C:\Users\PC\Documents\edley\aufgaben>python aufgabe8.py
Wir haben heute geschlossen. Wir wünschen einen schönen Sonntag.
C:\Users\PC\Documents\edley\aufgaben>~lock.aufgaben.odt#
```

Das Programm und der Screenshot zeigen die Ausführung an einem Sonntag, bei dem es sich nicht um einen Feiertag handelt. Für alle übrigen Kombinationen musst du einfach die Werte der beiden Variablen ändern.

## Lösung 9 | Vergleichsoperatoren

```
kontostand = 1254
```

```
betrag = int(input("Wie viel Geld möchten Sie abheben? "))
if betrag < kontostand:
    kontostand -= betrag
    print("Abhebung erfolgreich. Ihr neuer Kontostand beträgt:", kontostand)
elif betrag == kontostand:
    kontostand = 0
    print("Abhebung erfolgreich. Ihr Konto ist jetzt leer.")
else:
    print("Nicht genügend Guthaben vorhanden. Ihr Kontostand beträgt:",
kontostand)
```



```
Eingabeaufforderung
C:\Users\PC\Documents\edley\aufgaben>python aufgabe9.py
Wie viel Geld möchten Sie abheben? 400
Abhebung erfolgreich. Ihr neuer Kontostand beträgt: 854

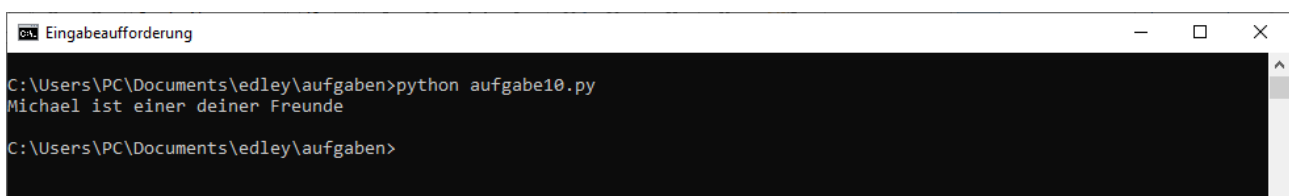
C:\Users\PC\Documents\edley\aufgaben>python aufgabe9.py
Wie viel Geld möchten Sie abheben? 1254
Abhebung erfolgreich. Ihr Konto ist jetzt leer.

C:\Users\PC\Documents\edley\aufgaben>python aufgabe9.py
Wie viel Geld möchten Sie abheben? 2000
Nicht genügend Guthaben vorhanden. Ihr Kontostand beträgt: 1254

C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 10 | Der in-Operator

```
freunde = ["Franziska", "Oliver", "Bettina", "Michael", "Joachim"]
if "Michael" in freunde:
    print("Michael ist einer deiner Freunde")
else:
    print("Du bist nicht mit Michael befreundet.")
```



```
Eingabeaufforderung
C:\Users\PC\Documents\edley\aufgaben>python aufgabe10.py
Michael ist einer deiner Freunde

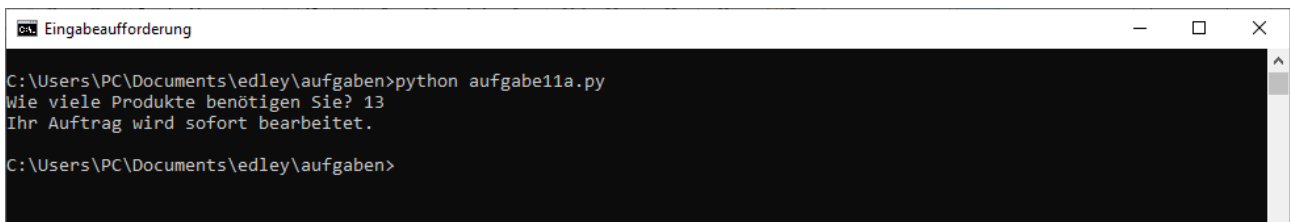
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 11 | Logische Operatoren – and, or, not

a)

```
bestand = 125
menge = int(input("Wie viele Produkte benötigen Sie? "))
mitarbeiter = True

if bestand >= menge and mitarbeiter:
    print("Ihr Auftrag wird sofort bearbeitet.")
else:
    print("Leider ist die Bearbeitung im Moment nicht möglich.")
```

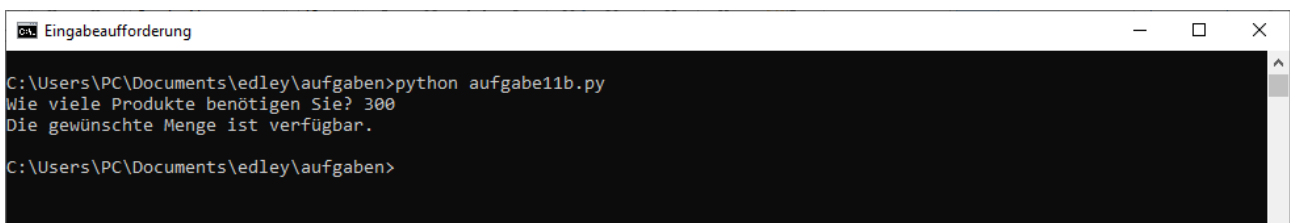


```
C:\Users\PC\Documents\edley\aufgaben>python aufgabe11a.py
Wie viele Produkte benötigen Sie? 13
Ihr Auftrag wird sofort bearbeitet.
C:\Users\PC\Documents\edley\aufgaben>
```

b)

```
bestandA = 125
bestandB = 321
menge = int(input("Wie viele Produkte benötigen Sie? "))
mitarbeiter = True

if bestandA >= menge or bestandB >= menge:
    print("Die gewünschte Menge ist verfügbar.")
else:
    print("Die gewünschte Menge ist nicht verfügbar.")
```



```
C:\Users\PC\Documents\edley\aufgaben>python aufgabe11b.py
Wie viele Produkte benötigen Sie? 300
Die gewünschte Menge ist verfügbar.
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 12 | Schleifen

a)

```
weiter = "ja"
liste = []

while weiter == "ja":
    zahl = int(input("Gib eine Zahl ein: "))
    liste.append(zahl)
    weiter = input("Möchtest du fortfahren? ")

print(liste)
```

```
C:\Users\PC\Documents\edley\aufgaben>python aufgabe12a.py
Gib eine Zahl ein: 4
Möchtest du fortfahren? ja
Gib eine Zahl ein: 7
Möchtest du fortfahren? ja
Gib eine Zahl ein: 123
Möchtest du fortfahren? ja
Gib eine Zahl ein: 9
Möchtest du fortfahren? nein
[4, 7, 123, 9]

C:\Users\PC\Documents\edley\aufgaben>
```

b)

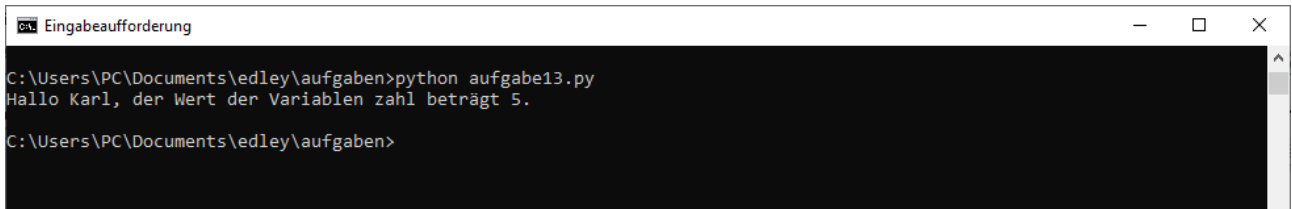
```
for i in range(1, 21):
    print(i*i)
```

```
C:\Users\PC\Documents\edley\aufgaben>python aufgabe12b.py
1
4
9
16
25
36
49
64
81
100
121
144
169
196
225
256
289
324
361
400

C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 13 | Stringformatierung

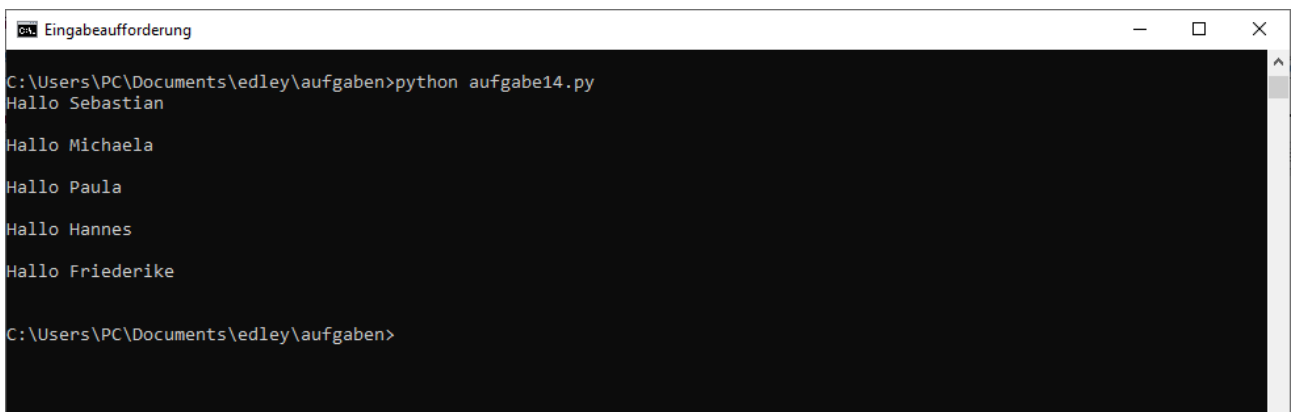
```
name = "Karl"  
zahl = 5  
print("Hallo %s, der Wert der Variablen zahl beträgt %i." %(name, zahl))
```



```
Eingabeaufforderung  
C:\Users\PC\Documents\edley\aufgaben>python aufgabe13.py  
Hallo Karl, der Wert der Variablen zahl beträgt 5.  
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 14 | Eingabe (Dateien lesen)

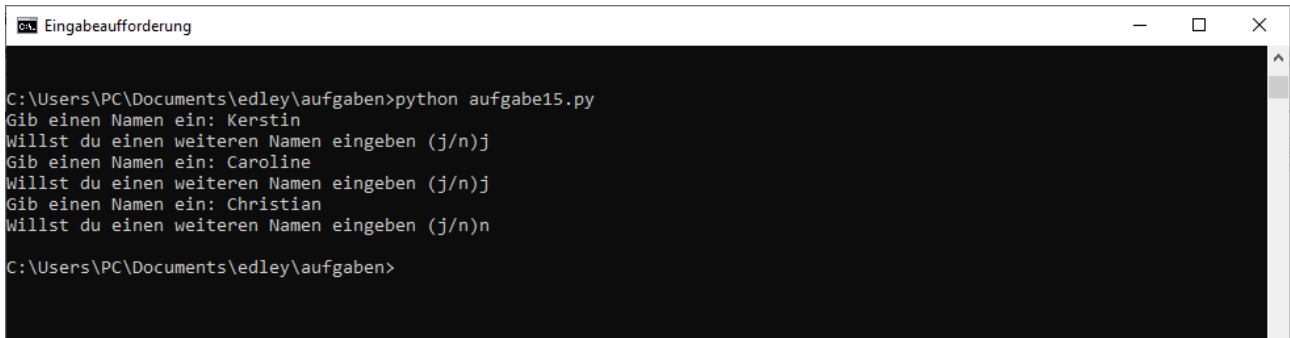
```
f = open("freunde.txt", "r")  
freunde = f.readlines()  
for name in freunde:  
    print("Hallo " + name)  
f.close()
```



```
Eingabeaufforderung  
C:\Users\PC\Documents\edley\aufgaben>python aufgabe14.py  
Hallo Sebastian  
Hallo Michaela  
Hallo Paula  
Hallo Hannes  
Hallo Friederike  
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 15 | Ausgabe (Dateien schreiben)

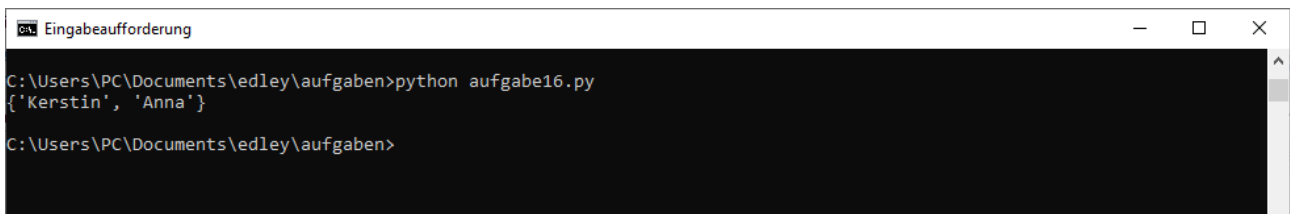
```
f = open("freunde.txt", "a")
weiter = "j"
while weiter == "j":
    name = input("Gib einen Namen ein: ")
    f.write(name + "\n")
    weiter = input("Willst du einen weiteren Namen eingeben (j/n)")
f.close()
```



```
C:\Users\PC\Documents\edley\aufgaben>python aufgabe15.py
Gib einen Namen ein: Kerstin
Willst du einen weiteren Namen eingeben (j/n)j
Gib einen Namen ein: Caroline
Willst du einen weiteren Namen eingeben (j/n)j
Gib einen Namen ein: Christian
Willst du einen weiteren Namen eingeben (j/n)n
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 16 | set und frozenset

```
freunde = {"Oliver", "Anna", "Michael"}
freunde.add("Kerstin")
freunde2 = {"Manuel", "Sabine", "Anna", "Kerstin", "Sebastian"}
print(freunde & freunde2)
```



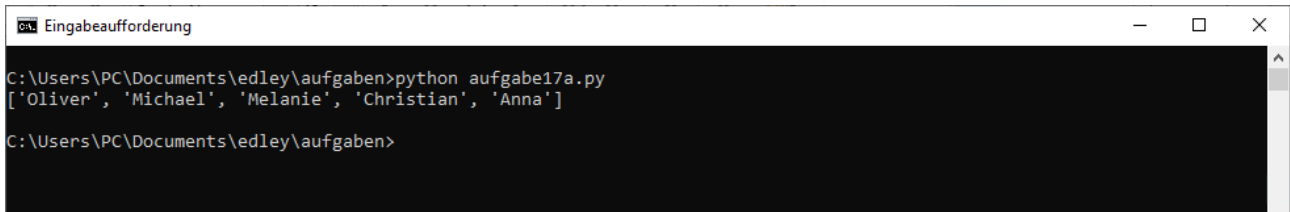
```
C:\Users\PC\Documents\edley\aufgaben>python aufgabe16.py
{'Kerstin', 'Anna'}
C:\Users\PC\Documents\edley\aufgaben>
```



## Lösung 17 | Sortierung (sort, sorted)

a)

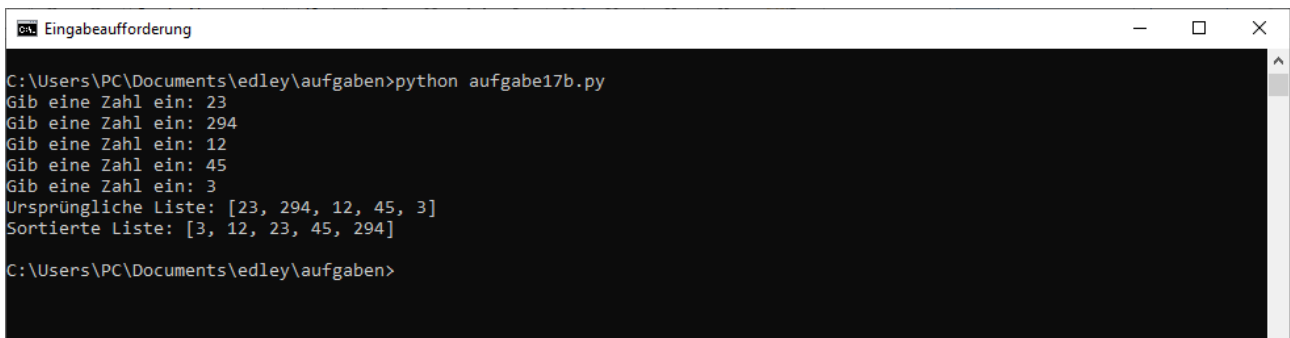
```
freunde = ["Oliver", "Christian", "Melanie", "Anna", "Michael"]
freunde.sort(reverse = True)
print(freunde)
```



```
C:\Users\PC\Documents\edley\aufgaben>python aufgabe17a.py
['Oliver', 'Michael', 'Melanie', 'Christian', 'Anna']
C:\Users\PC\Documents\edley\aufgaben>
```

b)

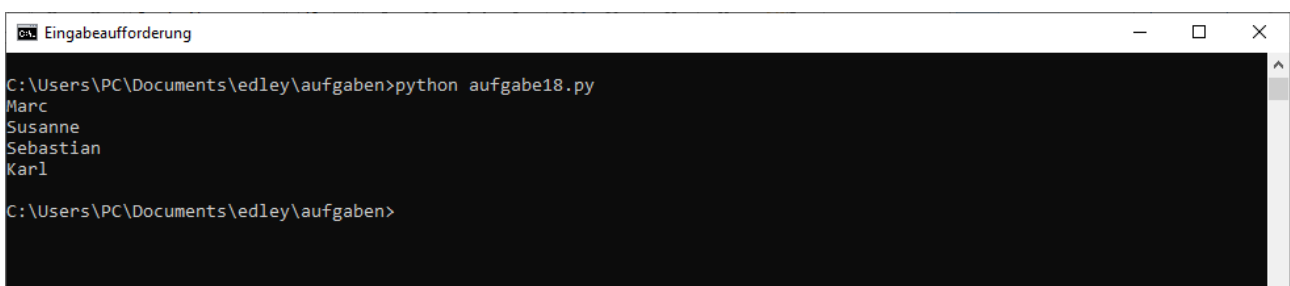
```
zahlen = []
for i in range (5):
    zahlen.append(int(input("Gib eine Zahl ein: ")))
zahlenSortiert = sorted(zahlen)
print("Ursprüngliche Liste:", zahlen)
print("Sortierte Liste:", zahlenSortiert)
```



```
C:\Users\PC\Documents\edley\aufgaben>python aufgabe17b.py
Gib eine Zahl ein: 23
Gib eine Zahl ein: 294
Gib eine Zahl ein: 12
Gib eine Zahl ein: 45
Gib eine Zahl ein: 3
Ursprüngliche Liste: [23, 294, 12, 45, 3]
Sortierte Liste: [3, 12, 23, 45, 294]
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 18 | Iteratoren

```
namen = {"Marc", "Susanne", "Sebastian", "Karl"}
iteration = iter(namen)
for i in range (4):
    print(next(iteration))
```



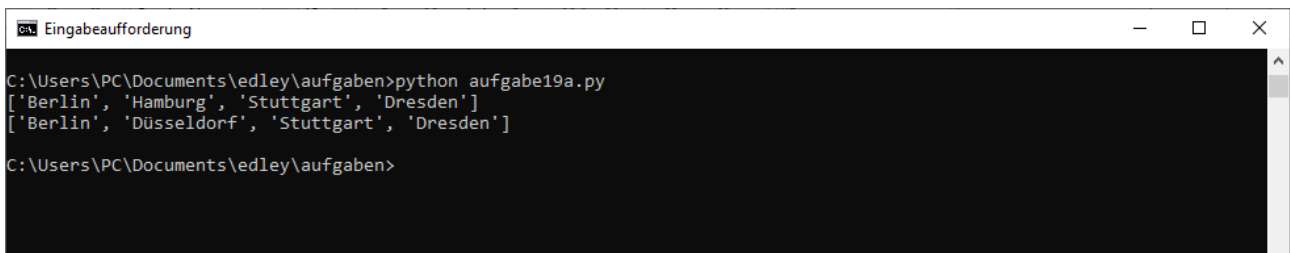
```
C:\Users\PC\Documents\edley\aufgaben>python aufgabe18.py
Marc
Susanne
Sebastian
Karl
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 19 | Copy und Deep Copy

a)

```
import copy
```

```
staedte = ["Berlin", "Hamburg", "Stuttgart", "Dresden"]
staedte2 = copy.copy(staedte)
staedte2[1] = "Düsseldorf"
print(staedte)
print(staedte2)
```

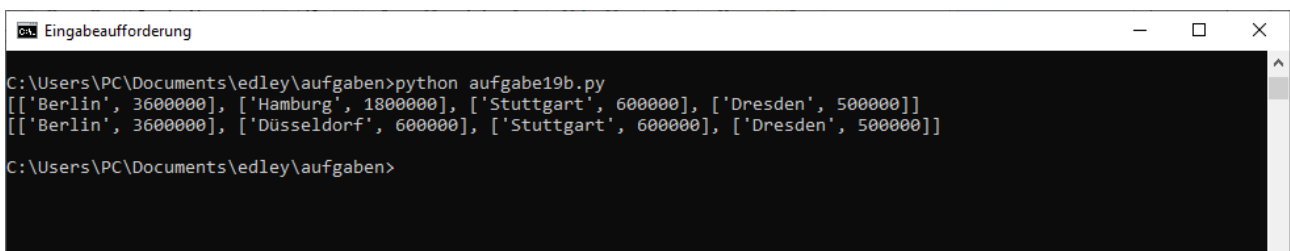


```
ca\ Eingabeaufforderung
C:\Users\PC\Documents\edley\aufgaben>python aufgabe19a.py
['Berlin', 'Hamburg', 'Stuttgart', 'Dresden']
['Berlin', 'Düsseldorf', 'Stuttgart', 'Dresden']
C:\Users\PC\Documents\edley\aufgaben>
```

b)

```
import copy
```

```
staedte = [["Berlin", 3600000], ["Hamburg", 1800000], ["Stuttgart", 600000],
["Dresden", 500000]]
staedte2 = copy.deepcopy(staedte)
staedte2[1][0] = "Düsseldorf"
staedte2[1][1] = 600000
print(staedte)
print(staedte2)
```

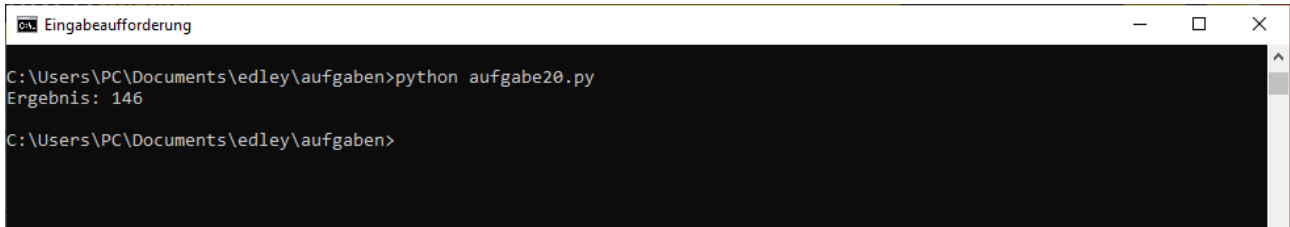


```
ca\ Eingabeaufforderung
C:\Users\PC\Documents\edley\aufgaben>python aufgabe19b.py
[['Berlin', 3600000], ['Hamburg', 1800000], ['Stuttgart', 600000], ['Dresden', 500000]]
[['Berlin', 3600000], ['Düsseldorf', 600000], ['Stuttgart', 600000], ['Dresden', 500000]]
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 20 | Funktionen selbst schreiben

```
def quadratischeGleichung(a, b, c, x):  
    print("Ergebnis:", a*x*x + b*x + c)
```

```
quadratischeGleichung(3, 6, 2, 6)
```



```
Eingabeaufforderung  
C:\Users\PC\Documents\edley\aufgaben>python aufgabe20.py  
Ergebnis: 146  
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 21 | Globale und lokale Variablen

```
def quadratischeGleichung():  
    print("Ergebnis:", a*x*x + b*x + c)
```

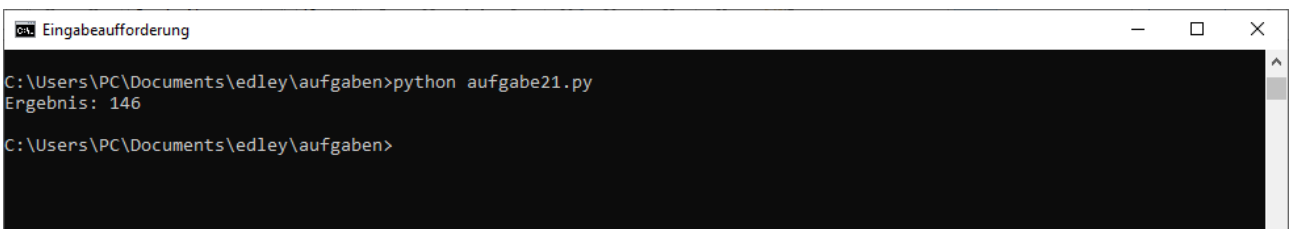
```
a = 3
```

```
b = 6
```

```
c = 2
```

```
x = 6
```

```
quadratischeGleichung()
```

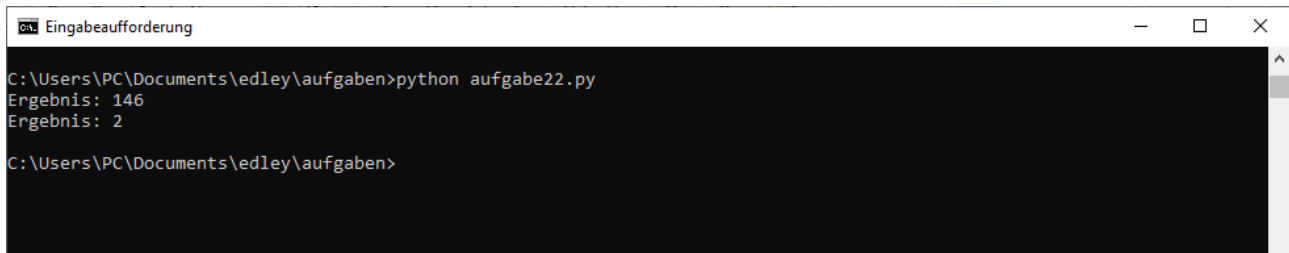


```
Eingabeaufforderung  
C:\Users\PC\Documents\edley\aufgaben>python aufgabe21.py  
Ergebnis: 146  
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 22 | Positionsparameter und Schlüsselwortparameter

```
def quadratischeGleichung(a, b, c, x = 0):  
    print("Ergebnis:", a*x*x + b*x + c)
```

```
quadratischeGleichung(3, 6, 2, 6)  
quadratischeGleichung(3, 6, 2)
```

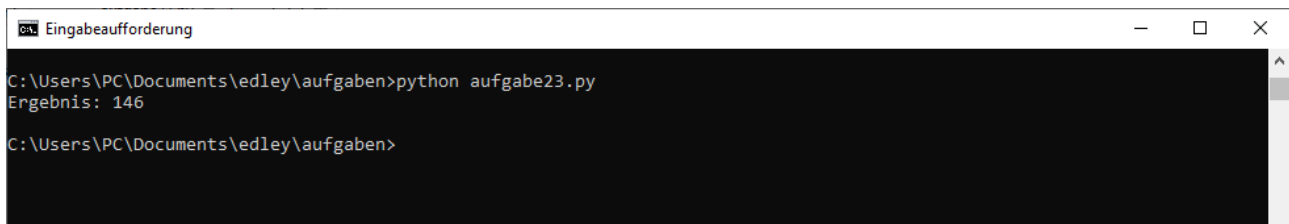


```
Eingabeaufforderung  
C:\Users\PC\Documents\edley\aufgaben>python aufgabe22.py  
Ergebnis: 146  
Ergebnis: 2  
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 23 | Rückgabewerte (return)

```
def quadratischeGleichung(a, b, c, x):  
    return a*x*x + b*x + c
```

```
ergebnis = quadratischeGleichung(3, 6, 2, 6)  
print("Ergebnis:", ergebnis)
```

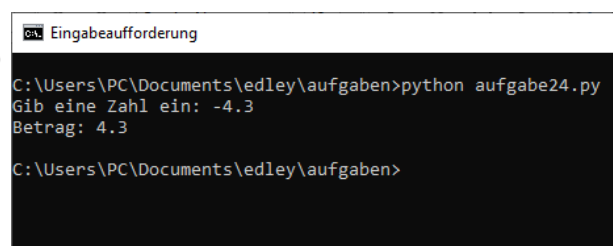


```
Eingabeaufforderung  
C:\Users\PC\Documents\edley\aufgaben>python aufgabe23.py  
Ergebnis: 146  
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 24 | Eingebaute Funktionen – Standardbibliothek

```
import math
```

```
zahl = float(input("Gib eine Zahl ein: "))  
print("Betrag:", math.fabs(zahl))
```



```
Eingabeaufforderung  
C:\Users\PC\Documents\edley\aufgaben>python aufgabe24.py  
Gib eine Zahl ein: -4.3  
Betrag: 4.3  
C:\Users\PC\Documents\edley\aufgaben>
```

## Lösung 25 | Objektorientierte Programmierung

```
class Besucher:
    """
    Erstellt das Objekt Besucher für ein Restaurant.
    """

    def __init__(self, tischnummer):
        """
        Initialisiert ein neues Objekt Besucher
        Argumente:
        * Tischnummer (int): Tischnummer
        """
        self.__Tischnummer = tischnummer
        self.__Bestellungen = []
        self.__Rechnungsbetrag = 0

    def bestellen (self, gericht):
        """
        Nimmt eine Bestellung auf.
        """
        self.__Bestellungen.append(gericht)
        self.__Rechnungsbetrag += Karte[gericht]

    def ausgeben(self):
        """
        Gibt alle Daten zu einem Gast aus.
        """
        print("\n\nTischnummer:", self.__Tischnummer)
        print("\nBestellte Speisen und Getränke:\n")
        for bestellung in self.__Bestellungen:
            print(bestellung)
        print("\n\nRechnungsbetrag:", self.__Rechnungsbetrag)
```

```
Karte = {"Schnitzel": 11.5, "Lasagne": 7.8, "Rinderfilet": 14.2, "Bier": 3.4,
"Wasser": 2.2}
neuerBesucher = Besucher(3)
neuerBesucher.bestellen("Schnitzel")
neuerBesucher.bestellen("Rinderfilet")
neuerBesucher.bestellen("Bier")
neuerBesucher.bestellen("Bier")
neuerBesucher.ausgeben()
```

```
Eingabeaufforderung
C:\Users\PC\Documents\edley\aufgaben>python aufgabe25.py

Tischnummer: 3
Bestellte Speisen und Getränke:
Schnitzel
Rinderfilet
Bier
Bier

Rechnungsbetrag: 32.5
C:\Users\PC\Documents\edley\aufgaben>
```