DS 1 – Weitere Übungen

Variablen, Eingabe und Ausgabe

Hier kannst du das Gelernte weiter üben. Die Punkte im Titel zeigen, wie schwer eine Aufgabe ist: ●○○ leicht, ●●○ mittel, ●●● knifflig. Fang mit den leichten an.

Die Aufgaben stehen auch im Skript im Abschnitt „Weitere Übungen“; die Lösungen im Lösungsheft.

Aufgabe 1: Mit oder ohne Anführungszeichen? ●○○

Lies das Programm, **ohne** es auszuführen. Schreibe auf, was es ausgibt. Überprüfe deine Vorhersage danach am Computer.

```
1 name = "Mia"
2 alter = 14
3 print("Name:", name)
4 print("alter")
5 print(alter)
6 print("In drei Jahren bist du", alter + 3)
```

```
Name: Mia
alter
14
In drei Jahren bist du 17
```

Deine Antwort:

…

Lösung

✅ **Lösung**

Lösung

Ausgabe:

```
Name: Mia
alter
14
In drei Jahren bist du 17
```

Lösung

In Zeile 4 steht `alter` in Anführungszeichen. Dann ist es ein Text, und genau dieser Text wird ausgegeben. In Zeile 5 fehlen die Anführungszeichen. Dann ist `alter` der Name einer Variablen, und Python gibt ihren Wert `14` aus. In Zeile 6 wird zuerst `14 + 3` ausgerechnet.

Aufgabe 2: Welcher Datentyp? ●○○

Das Programm wird mit den Eingaben `12`, `12` und `1.5` ausgeführt. Gib zu jeder Variablen an, welchen Datentyp ihr Wert hat (`int`, `float`, `str` oder `bool`). Überprüfe deine Antworten anschließend am Computer mit `print(type(...))`.

```python
a = input("Gib eine Zahl ein: ")
b = int(input("Gib eine Zahl ein: "))
c = float(input("Gib eine Zahl ein: "))
d = "3.5"
e = 3.5
f = False
```

```
Gib eine Zahl ein: 15
Gib eine Zahl ein: 15
Gib eine Zahl ein: 15
```

Variable	a	b	c	d	e	f
Datentyp						

Lösung

✅ Lösung

Lösung

Variable	a	b	c	d	e	f
Datentyp	`str`	`int`	`float`	`str`	`float`	`bool`

So kannst du es prüfen:

Lösung

```python
a = input("Gib eine Zahl ein: ")
b = int(input("Gib eine Zahl ein: "))
c = float(input("Gib eine Zahl ein: "))
d = "3.5"
e = 3.5
f = False
print(type(a), type(b), type(c))
print(type(d), type(e), type(f))
```

```
Gib eine Zahl ein: 15
Gib eine Zahl ein: 15
Gib eine Zahl ein: 15
<class 'str'> <class 'int'> <class 'float'>
<class 'str'> <class 'float'> <class 'bool'>
```

Lösung

Getestet mit den Eingaben: `12`, `12`, `1.5`

Lösung

Ausgabe:

```
Gib eine Zahl ein: «12»
Gib eine Zahl ein: «12»
Gib eine Zahl ein: «1.5»
<class 'str'> <class 'int'> <class 'float'>
<class 'str'> <class 'float'> <class 'bool'>
```

Lösung

`a` ist eine Zeichenkette, obwohl `12` eingegeben wurde: Eine Eingabe ist **immer** vom Typ `str`, solange man sie nicht umwandelt. `d` steht in Anführungszeichen und ist deshalb ebenfalls ein Text. `False` ohne Anführungszeichen ist ein Wahrheitswert.

Aufgabe 3: Kontostand verfolgen ●○○

Fülle das Ablaufprotokoll aus. Trage nach jedem Schritt die Werte der beiden Variablen ein (– bedeutet: noch kein Wert). Gib dann an, was das Programm ausgibt.

```
1 kontostand = 20
2 kontostand = kontostand - 5
3 taschengeld = 15
4 kontostand = kontostand + taschengeld
5 print("Kontostand:", kontostand)
```

```
Kontostand: 30
```

Schritt	Zeile	kontostand	taschengeld
1	1		
2	2		
3	3		
4	4		
5	5		

Deine Antwort:

…

Lösung

✅ **Lösung**

Lösung

Schritt	Zeile	kontostand	taschengeld
1	1	`20`	–
2	2	`15`	–
3	3	`15`	`15`
4	4	`30`	`15`
5	5	`30`	`15`

Lösung

Ausgabe:

```
Kontostand: 30
```

Lösung

In Zeile 2 wird zuerst rechts gerechnet (`20 - 5 = 15`), dann wird der alte Wert `20` überschrieben. In den Schritten 1 und 2 gibt es `taschengeld` noch nicht, denn die Variable bekommt erst in Zeile 3 einen Wert.

Aufgabe 4: Handyvertrag ●○○

Das Programm soll den monatlichen Preis eines Handyvertrags (Kommazahl) und die Laufzeit in Monaten (ganze Zahl) einlesen und die Gesamtkosten ausgeben, auf Cent gerundet. Ergänze die drei Lücken `___` und teste dein Programm mit den Eingaben `9.99` und `24`.

Lückencode

```
preis = ___(input("Preis pro Monat in Euro: "))
monate = ___(input("Laufzeit in Monaten: "))
gesamt = preis * monate
print("Gesamtkosten:", ___, "Euro")
```

Lösung

✅ **Lösung**

Lösung

```
preis = float(input("Preis pro Monat in Euro: "))
monate = int(input("Laufzeit in Monaten: "))
gesamt = preis * monate
print("Gesamtkosten:", round(gesamt, 2), "Euro")
```

```
Preis pro Monat in Euro: 12
Laufzeit in Monaten: 12
Gesamtkosten: 144.0 Euro
```

Lösung

Getestet mit den Eingaben: `9.99`, `24`

Lösung

Ausgabe:

```
Preis pro Monat in Euro: «9.99»
Laufzeit in Monaten: «24»
Gesamtkosten: 239.76 Euro
```

Lösung

Der Preis hat Nachkommastellen, deshalb `float`. Die Anzahl der Monate ist eine ganze Zahl, deshalb `int`. `round(gesamt, 2)` rundet auf zwei Nachkommastellen, also auf Cent. Ohne `round` kann bei manchen Eingaben eine lange Kommazahl erscheinen.

Aufgabe 5: Fehler finden: Temperatur umrechnen ●●○

In den USA misst man Temperaturen in Grad Fahrenheit. Man rechnet so um: Grad Fahrenheit = Grad Celsius · 1,8 + 32. Das Programm soll eine Temperatur in Grad Celsius einlesen und in Grad Fahrenheit umrechnen. Es enthält **drei Fehler**.

```
celsius = input("Temperatur in Grad Celsius: ")
fahrenheit = celsius * 1.8 + 32
print("Das sind", Fahrenheit, "Grad Fahrenheit.)
```

```
  File "<Zelle 7>", line 3
    print("Das sind", Fahrenheit, "Grad Fahrenheit.)
                                  ^
SyntaxError: unterminated string literal (detected at line 3)
```

1. Finde die drei Fehler. Notiere jeweils die Zeile und was falsch ist.
2. Schreibe das Programm richtig auf und teste es mit `20` (richtig ist 68 Grad Fahrenheit).

Deine Antwort:

…

Lösung

✅ **Lösung**

Lösung

1. **Zeile 1:** Die Eingabe wird nicht umgewandelt. `celsius` ist ein Text, und mit einem Text kann man nicht rechnen. Python meldet in Zeile 2 einen `TypeError`. Richtig: `float(input(...))`.
2. **Zeile 3:** `Fahrenheit` mit großem F. Die Variable heißt aber `fahrenheit`. Python meldet einen `NameError`.
3. **Zeile 3:** Nach `Grad Fahrenheit.` fehlt das schließende Anführungszeichen. Python meldet einen `SyntaxError`.

Lösung

```
1 celsius = float(input("Temperatur in Grad Celsius: "))
2 fahrenheit = celsius * 1.8 + 32
3 print("Das sind", fahrenheit, "Grad Fahrenheit.")
```

```
Temperatur in Grad Celsius: 23
Das sind 73.4 Grad Fahrenheit.
```

Lösung

Getestet mit den Eingaben: `20`

Lösung

Ausgabe:

```
Temperatur in Grad Celsius: «20»
Das sind 68.0 Grad Fahrenheit.
```

Lösung

Python meldet zuerst den `SyntaxError`, denn die Schreibweise prüft es, bevor das Programm überhaupt startet. Die anderen beiden Fehler zeigen sich erst, wenn das Programm läuft. Auch `int` statt `float` ist richtig, dann kann man aber keine Kommazahlen eingeben.

Aufgabe 6: Taschengeld ●●○

Schreibe ein Programm, das das wöchentliche Taschengeld (Kommazahl) und eine Anzahl von Wochen (ganze Zahl) einliest. Es soll ausgeben, wie viel Geld in dieser Zeit zusammenkommt. Danach soll es den Preis eines Wunschartikels einlesen und ausgeben, wie viel Geld nach dem Kauf noch übrig ist. Runde die Beträge auf Cent.

So könnte es aussehen:

Ausgabe:

```
Taschengeld pro Woche in Euro: «7.5»
Anzahl der Wochen: «6»
In 6 Wochen bekommst du 45.0 Euro.
Preis des Wunschartikels in Euro: «29.99»
Nach dem Kauf bleiben 15.01 Euro übrig.
```

Lösung

✅ **Lösung**

Lösung

```
1 taschengeld = float(input("Taschengeld pro Woche in Euro: "))
2 wochen = int(input("Anzahl der Wochen: "))
3 gespart = taschengeld * wochen
4 print("In", wochen, "Wochen bekommst du", round(gespart, 2), "Euro.")
5 preis = float(input("Preis des Wunschartikels in Euro: "))
6 rest = gespart - preis
7 print("Nach dem Kauf bleiben", round(rest, 2), "Euro übrig.")
```

```
Taschengeld pro Woche in Euro: 12
Anzahl der Wochen: 4
In 4 Wochen bekommst du 48.0 Euro.
Preis des Wunschartikels in Euro: 12
Nach dem Kauf bleiben 36.0 Euro übrig.
```

Lösung

Getestet mit den Eingaben: `7.5`, `6`, `29.99`

Lösung

Ausgabe:

```
Taschengeld pro Woche in Euro: «7.5»
Anzahl der Wochen: «6»
In 6 Wochen bekommst du 45.0 Euro.
Preis des Wunschartikels in Euro: «29.99»
Nach dem Kauf bleiben 15.01 Euro übrig.
```

Lösung

Das Zwischenergebnis wird in `gespart` gespeichert, damit man es in Zeile 6 weiterverwenden kann. Gerade bei `45.0 - 29.99` ist `round` wichtig: Ohne Runden zeigt Python hier `15.010000000000002` an. Das liegt daran, dass der Computer Kommazahlen nicht immer ganz exakt speichert.

Aufgabe 7: Pfannkuchen für alle ●●○

Für **eine** Person braucht man für Pfannkuchen 62,5 g Mehl, 0,5 Eier und 125 ml Milch.

1. Schreibe ein Programm, das die Anzahl der Personen einliest und ausgibt, wie viel man von jeder Zutat braucht.
2. Begründe, warum du die Anzahl der Personen mit `int` und nicht mit `float` umwandelst.

Deine Antwort:

…

Lösung

✅ **Lösung**

Lösung

```python
personen = int(input("Für wie viele Personen? "))
mehl = 62.5 * personen
eier = 0.5 * personen
milch = 125 * personen
print("Mehl:", mehl, "g")
print("Eier:", eier)
print("Milch:", milch, "ml")
```

```
Für wie viele Personen? 4
Mehl: 250.0 g
Eier: 2.0
Milch: 500 ml
```

Lösung

Getestet mit den Eingaben: `6`

Lösung

Ausgabe:

```
Für wie viele Personen? «6»
Mehl: 375.0 g
Eier: 3.0
Milch: 750 ml
```

Lösung

2. Personen kann man nur ganz zählen. Eine Eingabe wie `2.5` wäre unsinnig, und `int` würde sie mit einem `ValueError` ablehnen. Die Mengen für Mehl und Eier sind trotzdem Kommazahlen, weil mit `62.5` und `0.5` gerechnet wird. Bei der Milch wird nur mit ganzen Zahlen gerechnet, deshalb steht dort `750` ohne `.0`.

Aufgabe 8: Busfahrt oder Monatskarte? ●●●

Eine Einzelfahrt mit dem Bus kostet 2,10 Euro. Eine Monatskarte kostet 58 Euro.

1. Schreibe ein Programm, das einliest, wie viele Busfahrten man an einem Schultag macht und an wie vielen Schultagen man im Monat fährt. Es soll ausgeben, was die Einzelfahrten im Monat zusammen kosten. Danach soll es ausgeben, wie viel man mit der Monatskarte spart, also die Kosten der Einzelfahrten minus 58 Euro.
2. Teste dein Programm mit `2` Fahrten an `20` Tagen und mit `2` Fahrten an `10` Tagen. Erkläre, was ein **negatives** Ergebnis für die Ersparnis bedeutet.

Deine Antwort:

…

Lösung

✅ Lösung

Lösung

```
fahrten_pro_tag = int(input("Busfahrten pro Schultag: "))
tage = int(input("Schultage im Monat: "))
kosten_einzeln = fahrten_pro_tag * tage * 2.10
ersparnis = kosten_einzeln - 58
print("Einzelfahrten kosten:", round(kosten_einzeln, 2), "Euro")
print("Ersparnis mit Monatskarte:", round(ersparnis, 2), "Euro")
```

```
Busfahrten pro Schultag: 2
Schultage im Monat: 20
Einzelfahrten kosten: 84.0 Euro
Ersparnis mit Monatskarte: 26.0 Euro
```

Lösung

Getestet mit den Eingaben: `2`, `20`

Lösung

Ausgabe:

```
Busfahrten pro Schultag: «2»
Schultage im Monat: «20»
Einzelfahrten kosten: 84.0 Euro
Ersparnis mit Monatskarte: 26.0 Euro
```

Lösung

Mit `2` und `10` kosten die Einzelfahrten 42 Euro, die Ersparnis ist `-16.0` Euro. Ein negatives Ergebnis bedeutet: Die Monatskarte ist **teurer** als die Einzelfahrten, man würde 16 Euro mehr bezahlen. Dann lohnt sie sich nicht.

Wichtig ist die Reihenfolge der Rechnung: Erst werden die Kosten der Einzelfahrten berechnet und gespeichert, dann wird mit diesem Wert weitergerechnet. Ob sich die Monatskarte lohnt, kann das Programm selbst noch nicht entscheiden. Das lernst du im Abschnitt über Verzweigungen.