

Lektion 4a/4b – Musterlösungen

Hinweis: Bis einschließlich Lektion 4a werden keine Schleifen verwendet.

Lektion 4a – Aufgabe 1 (Fragen) – Lösungen

- 1) `right(90)` dreht die Turtle um 90° nach rechts.
- 2) `forward(100)` bewegt die Turtle 100 Einheiten nach vorne.
- 3) Füllbereich: Alles zwischen `begin_fill()` und `end_fill()` wird gefüllt; wird `end_fill()` zu früh gesetzt, füllt es nur den bis dahin gezeichneten Teil.
- 4) Mit `right(45)` sind die Ecken kleiner; bei vier Seiten entsteht kein Quadrat (mehr Seiten nötig).
- 5) `backward(...)` bewegt zurück; `left(...)` dreht nach links.
- 6) `def` definiert eine Funktion (Name und Codeblock).
- 7) Falsche Einrückungen führen zu Fehlern (z. B. `IndentationError`).

Aufgabe 2: Erstellen Sie weitere Formen

```
from turtle import *
```

```
#### Aufgabe 2: Weitere Formen zeichnen
```

```
# 1. Benennen Sie die Funktion in square() um (statt mystery) und ändern Sie:
```

```
# a) die Farbe der Linien
```

```
# b) die Füllfarbe
```

```
# c) die Länge der Linien
```

```
def square():  
    fillcolor("yellow")  
    begin_fill()  
    pencolor("blue")  
    forward(80)  
    right(90)  
    forward(80)  
    right(90)  
    forward(80)  
    right(90)  
    forward(80)  
    end_fill()
```

```
square()
```

2. Erstellen Sie eine neue Funktion mit dem Namen triangle(). Sie müssen den Winkel zwischen den Linien anpassen.

```
def triangle():  
    forward(150)  
    right(120)  
    forward(150)  
    right(120)  
    forward(150)  
    right(120)
```

```
clear()  
triangle()
```

3. Verwenden Sie penup() und pendown(), um ein Quadrat und ein Dreieck mit einem Abstand dazwischen zu zeichnen.

```
def drawing():  
    square()  
    penup()  
    forward(50)  
    pendown()  
    triangle()
```

```
clear()  
drawing()
```

4. Verwenden Sie die Funktion circle(), um zwei Kreise zu zeichnen –einen innerhalb des anderen.

```
def my_circles():  
    circle(50)  
    pendown()  
    circle(40)
```

```
clear()  
my_circles()
```

5. Zeichnen Sie die Form auf der rechten Seite (mehrere Dreiecke, die sich überlappen).

```
def triangle():  
    for counter in range(3):  
        forward(100)  
        right(120)
```

```
def shape():  
    for counter in range(6):  
        triangle()  
        left(60)
```

```
clear()  
shape()
```

Lektion 4b: Iteration – Übungen mit Turtles

```
from turtle import *
```

1. Ändern Sie Ihre triangle()-Funktion so, dass sie eine for-Schleife verwendet.

```
def triangle():  
    for counter in range(3):  
        forward(150)  
        right(120)
```

```
triangle()
```

2. Schreiben Sie eine Funktion pentagon(), die ein Fünfeck (5 Seiten)

mit einer Seitenlänge von 200 zeichnet.

```
def pentagon():  
    angle = 360.0 / 5  
    for counter in range(5):  
        forward(200)  
        right(angle)
```

```
clear()  
pentagon()
```

3. Schreiben Sie eine Funktion hexagon(), die ein blaues Sechseck (6 Seiten)

mit einer Seitenlänge von 50 zeichnet.

```
def hexagon():  
    angle = 360 / 6  
    fillcolor("blue")  
    begin_fill()  
    for counter in range(6):  
        forward(50)  
        right(angle)  
    end_fill()
```

```
clear()  
hexagon()
```

```
from turtle import *
```

```
#### Erweiterungsaufgabe
```

```
#### Schreiben Sie eine Funktion envelope(), die einen Umschlag zeichnet.
```

```
#### Hinweis: Die Funktion ruft square() und triangle() auf.
```

```
def square():
```

```
    # Zeichnet ein Quadrat mit Seitenlänge 150
```

```
    for counter in range(4):
```

```
        forward(150)
```

```
        right(90)
```

```
def triangle():
```

```
    # Zeichnet ein gleichseitiges Dreieck mit Seitenlänge 150
```

```
    for counter in range(3):
```

```
        forward(150)
```

```
        right(120)
```

```
def envelope():
```

```
    # Zeichnet einen Umschlag, indem Quadrat und Dreieck kombiniert werden
```

```
    square()
```

```
    triangle()
```

```
# Aufruf der Funktion, um den Umschlag zu zeichnen
```

```
envelope()
```

Schlüsselbegriffe & Hinweise

Iteration: Wiederholung mit for (feste Anzahl) oder while (Bedingung). Beispiel: for i in range(n): ...

Wichtige Punkte: range(start, stop, step) – obere Grenze ist exklusiv.

Vertiefung: Zahlenfolgen

```
zahl = 2
for i in range(0, 5):
    zahl = zahl * 3
print("Das Ergebnis lautet", zahl)
```

Weitere Beispiele: range(3, 10, 2) → 3,5,7,9 ; range(10, 0, -1) → 10..1

Beispielprogramme

Zahlen 1..10 ausgeben

```
for i in range(1, 11):
    print(i)
```

Summe 1..100 berechnen

```
summe = 0
for i in range(1, 101):
    summe += i
print(summe) # 5050
```

Übungsaufgaben – Lösungen

Quadratzahlen 0..10

```
for zahl in range(0, 11):
    print(f"{zahl} * {zahl} = {zahl ** 2}")
```

Zahlensumme (N Eingaben addieren)

```
anzahl = int(input("Wie viele Zahlen möchtest du addieren? "))
summe = 0
for i in range(anzahl):
    summe += int(input(f"Gib die {i+1}. Zahl an: "))
print(f"Das Ergebnis lautet: {summe}")
```

Summe nur aus positiven Zahlen

```
anzahl = int(input("Wie viele Zahlen möchtest du eingeben? "))
summe = 0
for i in range(anzahl):
    zahl = int(input(f"Gib die {i+1}. Zahl an: "))
    if zahl > 0:
        summe += zahl
print(f"Die Summe der positiven Zahlen lautet: {summe}")
```