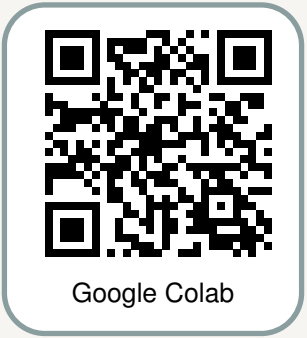
Python Grundlagen

Kontrollstrukturen


Google Colab

Lernziele

Am Ende dieser Lektion können Sie:

- ✓ Bedingte Anweisungen (if/elif/else)
- ✓ Vergleichs- und logische Operatoren
- ✓ Schleifen (for, while)
- ✓ Sprunganweisungen (break, continue)

Was sind Kontrollstrukturen?

Kontrollstrukturen steuern den Programmablauf:

- **Bedingte Anweisungen**: Entscheidungen treffen (if/else)
- **Schleifen**: Wiederholungen (for, while)
- **Sprunganweisungen**: Ablauf unterbrechen (break, continue)

Die if-Anweisung

Führt Code aus, wenn eine Bedingung erfüllt ist:

```
1 alter = 19
2
3 if alter >= 18:
4     print("Du bist volljährig.")
```

Wichtig: Einrückung (Tab/4 Spaces) ist in Python Pflicht!

if-else: Entscheidungen treffen

```
alter = 16

if alter >= 18:
    print("Du bist volljährig.")
else:
    print("Du bist noch nicht volljährig.")
```

Der else-Block wird ausgeführt, wenn die if-Bedingung **nicht** erfüllt ist.

if-elif-else: Mehrere Bedingungen

```
note = 2.8

if note <= 1.5:
    print("Sehr gut!")
elif note <= 2.5:
    print("Gut!")
elif note <= 3.5:
    print("Befriedigend!")
elif note <= 4.0:
    print("Ausreichend!")
else:
    print("Nicht bestanden!")
```

Vergleichsoperatoren

==	gleich
!=	ungleich
>	größer als
<	kleiner als
>=	größer oder gleich
<=	kleiner oder gleich

Logische Operatoren

```
alter = 25
einkommen = 2000

if alter >= 18 and einkommen > 1500:
    print("Beide Bedingungen erf llt.")

if alter > 16 or einkommen < 1000:
    print("Mindestens eine Bedingung erf llt.")

if not alter < 18:
    print("Du bist nicht minderj hrig.")
```

Mini-Übung 1

Schreibe ein Programm, das prüft:

- Wenn die Zahl größer als 0 ist: "Positiv" ausgeben
- Wenn die Zahl kleiner als 0 ist: "Negativ" ausgeben
- Sonst: "Null" ausgeben

```
zahl = int(input("Zahl eingeben:  "))
```

Lösung: Mini-Übung 1

```
zahl = int(input("Zahl eingeben: "))

if zahl > 0:
    print("Positiv")
elif zahl < 0:
    print("Negativ")
else:
    print("Null")
```

Die for-Schleife

Wiederholt Code für jedes Element einer Sequenz:

```
1 fruechte = ["Apfel", "Banane", "Kirsche"]
2
3 for frucht in fruechte:
4     print(frucht)
```

Ausgabe:

```
Apfel
Banane
Kirsche
```

Die range()-Funktion

```
# range(stop): 0 bis 4
for i in range(5):
    print(i)  # 0, 1, 2, 3, 4

# range(start, stop): 1 bis 6
for i in range(1, 7):
    print(i)  # 1, 2, 3, 4, 5, 6

# range(start, stop, step): 2, 4, 6, 8, 10
for i in range(2, 11, 2):
    print(i)
```

Die while-Schleife

Wiederholt Code, solange eine Bedingung erfüllt ist:

```
1 zaehler = 0
2
3 while zaehler < 5:
4     print(zaehler)
5     zaehler += 1   # Wichtig: Z hler erh hen!
```

Vergiss das `zaehler += 1` nicht! Sonst: **Endlosschleife**!

break und continue

```
# break: Schleife sofort beenden
for i in range(1, 11):
    if i == 5:
        break       # Beendet bei 5
    print(i)        # Gibt 1, 2, 3, 4 aus

# continue: Aktuellen Durchlauf   berspringen
for i in range(1, 6):
    if i == 3:
        continue    #   berspringt   3
    print(i)        # Gibt 1, 2, 4, 5 aus
```

Verschachtelte Schleifen

```
for i in range(1, 4):
    for j in range(1, 4):
        print(i, "x", j, "=", i*j)
    print()  # Leere Zeile nach jeder Reihe
```

Erzeugt eine kleine Multiplikationstabelle (1x1 bis 3x3)

Häufige Fehler

```
# Fehler 1: = statt ==
if x = 5:          # SyntaxError!
    print("x ist 5")

# Fehler 2: Einr ckung vergessen
if x == 5:
print("x ist 5")   # IndentationError!

# Fehler 3: Doppelpunkt vergessen
if x == 5          # SyntaxError!
    print("x ist 5")
```

Übung: Fehlersuche

```
zahl = 10
if zahl % 2 = 0:
    print("Gerade")

summe = 0
for i in range(1, 6)
    summe += i
print(summe)
```

Lösung: Fehlersuche

```
zahl = 10
if zahl % 2 == 0:          # == statt =
    print("Gerade")

summe = 0
for i in range(1, 6):      # Doppelpunkt fehlte
    summe += i
print(summe)
```

Klausurvorbereitung: Was gibt dieser Code aus?

```
for i in range(5):
    if i == 2:
        continue
    if i == 4:
        break
    print(i)
```

Tipp: `continue` überspringt den Rest, `break` beendet die Schleife.

Lösung

```
0
1
3
```

- 0 und 1 werden normal ausgegeben
- Bei 2: `continue` überspringt die Ausgabe
- 3 wird ausgegeben
- Bei 4: `break` beendet die Schleife

Zusammenfassung

- **if-elif-else**: Bedingte Ausführung
- **Vergleiche**: `==`, `!=`, `>`, `<`, `>=`, `<=`
- **Logik**: `and`, `or`, `not`
- **Schleifen**: `for` (Sequenzen), `while` (Bedingung)
- **Sprünge**: `break` , `continue`

Hausaufgaben

1. Schreibe ein Programm, das alle geraden Zahlen von 1 bis 20 ausgibt
2. Erstelle einen einfachen Taschenrechner mit if-elif-else
3. Schreibe ein Programm, das ein Passwort auf 8 Zeichen Länge prüft