DS 3 – Verzweigungen: Programme treffen Entscheidungen

Informatik 10 – MINT-Zweig | Textuelle Programmierung und Datenstrukturen

Lernziele

Ich kann

- erklären, woraus eine Verzweigung besteht, und die Begriffe *Bedingung*, *dann-Teil* und *sonst-Teil* verwenden,
- ein-, zwei- und mehrseitige Verzweigungen mit `if`, `else` und `elif` schreiben,
- mit der Einrückung festlegen, welche Anweisungen zu einem Block gehören, und Einrückungsfehler beheben,
- den Ablauf eines Programms mit Verzweigung in einem Ablaufprotokoll darstellen,
- Bedingungen mit `and` und `or` formulieren und Verzweigungen verschachteln.

🗣️ Sprachhilfe

- *„Ich bekomme raus, ob die Zahl größer als 4 ist.“*
- → „Die Anweisung gibt ‚Die Zahl ist größer als 4.‘ aus, wenn die Zahl größer als 4 ist, und sonst ‚Die Zahl ist nicht größer als 4.‘“
- → „Die Alternativanweisung prüft, ob eine Zahl größer als 4 ist. Ist dies der Fall, wird der dann-Teil ausgeführt, sonst der sonst-Teil.“

Nützliche Wendungen: *Die Verzweigung besteht aus … · Es wird geprüft, ob … · Ist die Bedingung erfüllt, wird …*

Worum es geht

Wenn es regnet, *dann* nehme ich einen Schirm mit. *Wenn* der Bus kommt, steige ich ein, *sonst* warte ich. Bisher hat Python jede Zeile ausgeführt, von oben nach unten. Jetzt lernst du, wie ein Programm abhängig von einer **Bedingung** nur bestimmte Anweisungen ausführt. Aus der Blockprogrammierung kennst du „falls … dann“ und „falls … dann … sonst“.

1 Die einseitige Verzweigung: `if`

```
1 temperatur = 30
2 if temperatur > 25:
3     print("Es ist heiß. Trink genug Wasser!")
4 print("Schönen Tag!")
```

```
Es ist heiß. Trink genug Wasser!
Schönen Tag!
```

- **Zeile 2:** `if` heißt „wenn“ oder „falls“. Python prüft die Bedingung `temperatur > 25`: 30 > 25 ist `True`. Am Ende steht ein **Doppelpunkt**.
- **Zeile 3:** Diese Zeile ist **eingerückt** (vier Leerzeichen). Sie gehört zur `if`-Anweisung und wird nur ausgeführt, weil die Bedingung wahr ist.
- **Zeile 4:** Nicht eingerückt, wird also in jedem Fall ausgeführt.

🔮 **Vorhersage:** Was gibt das Programm bei `temperatur = 18` aus? Überlege zuerst, dann führe die Zelle aus.

```python
temperatur = 18
if temperatur > 25:
    print("Es ist heiß. Trink genug Wasser!")
print("Schönen Tag!")
```

```
Schönen Tag!
```

Die Bedingung ist `False`, Python **überspringt** Zeile 3. Anweisungen wie `if`, die den Ablauf steuern, heißen **Kontrollstrukturen**.

> 📌 **Definition:** Eine **Bedingung** ist ein Ausdruck, dessen Wert `True` oder `False` ist, z. B. `temperatur > 25`. Eine **Verzweigung** (bedingte Anweisung) ist eine Kontrollstruktur, die abhängig von einer Bedingung festlegt, welche Anweisungen ausgeführt werden.

> 📌 **Merksatz: Einseitige Verzweigung:** `if Bedingung:` – der eingerückte Block wird nur ausgeführt, wenn die Bedingung `True` ist.

2 Einrückung: Was gehört zum Block?

Alle Zeilen direkt unter dem `if`, die gleich weit eingerückt sind, gehören zum Block. 🔮 **Vorhersage:** Die beiden Programme unterscheiden sich nur in Zeile 4. Was gibt jedes aus?

```python
punkte = 3
if punkte >= 10:
    print("Gewonnen!")
    print("Du bekommst einen Preis.")
print("Spiel beendet.")
```

```
Spiel beendet.
```

```python
punkte = 3
if punkte >= 10:
    print("Gewonnen!")
print("Du bekommst einen Preis.")
print("Spiel beendet.")
```

```
Du bekommst einen Preis.
Spiel beendet.
```

Im ersten Programm werden Zeilen 3 und 4 übersprungen. Im zweiten gehört nur Zeile 3 zum Block, und das Programm verteilt einen Preis, obwohl niemand gewonnen hat.

> 📌 **Merksatz:** Die **Einrückung** (4 Leerzeichen) bestimmt, welche Anweisungen zu einem Block gehören. Die Zeile mit der Bedingung endet mit einem **Doppelpunkt** `:`.

⚠️ Stolpersteine: Doppelpunkt und Einrückung

Führe die drei Zellen aus und lies jeweils die letzte Zeile der Fehlermeldung.

```
1 zahl = 5
2 if zahl > 0
3     print("positiv")
```

```
  File "<Zelle 5>", line 2
    if zahl > 0
               ^
SyntaxError: expected ':'
```

```
1 zahl = 5
2 if zahl > 0:
3 print("Die Zahl ist positiv.")
```

```
  File "<Zelle 6>", line 3
    print("Die Zahl ist positiv.")
    ^^^^^
IndentationError: expected an indented block after 'if' statement on line 2
```

```
zahl = 5
    print(zahl)
```

```
  File "<Zelle 7>", line 2
    print(zahl)
IndentationError: unexpected indent
```

1. `SyntaxError: expected ':'`: Der Doppelpunkt fehlt, `^` zeigt auf die Stelle.
2. `IndentationError: expected an indented block after 'if' statement on line 2`: Nach dem `if` fehlt der eingerückte Block.
3. `IndentationError: unexpected indent`: Eine Zeile ist eingerückt, obwohl kein Block mit Doppelpunkt davor steht.

> 💡 **Tipp:** Nach einem Doppelpunkt rückt die Umgebung meist automatisch ein. Achte darauf, dass alle Zeilen eines Blocks **genau gleich weit** eingerückt sind.

3 Die zweiseitige Verzweigung: `if` … `else`

Soll das Programm in **beiden** Fällen etwas tun, gibt es `else` („sonst“).

```
1 zahl = int(input("Gib eine Zahl ein: "))
2 if zahl > 4:
3     print("Die Zahl ist größer als 4.")
4 else:
5     print("Die Zahl ist nicht größer als 4.")
6 print("Fertig.")
```

```
Gib eine Zahl ein: 15
Die Zahl ist größer als 4.
Fertig.
```

- **Zeile 3:** der **dann-Teil**, wird ausgeführt, wenn die Bedingung wahr ist.
- **Zeile 4:** `else:` steht genau unter dem `if`, hat **keine** eigene Bedingung, aber einen Doppelpunkt.
- **Zeile 5:** der **sonst-Teil**, nur wenn die Bedingung falsch ist.

Es wird **immer genau einer** der beiden Teile ausgeführt. Deshalb heißt die zweiseitige Verzweigung auch **Alternativanweisung**.

Das **Ablaufprotokoll** zeigt, welche Zeilen wirklich ausgeführt werden, links für die Eingabe 7, rechts für 2:

Schritt	Zeile	zahl		Schritt	Zeile	zahl
1	1	7		1	1	2
2	2	7		2	2	2
3	3	7		3	5	2
4	6	7		4	6	2

Die Zeile `else:` selbst tut nichts, deshalb taucht Zeile 4 nicht auf.

> 📌 **Merksatz: Zweiseitige Verzweigung:** `if … else:` – entweder der **dann-Teil** oder der **sonst-Teil** wird ausgeführt.

🔮 Übung 1 – Glatteis-Warnung

Lies das Programm. Beschreibe in zwei bis drei Sätzen mit Fachbegriffen, was es tut. Gib dann **ohne Ausführen** an, was es bei `-3`, `0` und `5` ausgibt. Überprüfe danach.

```
1 temperatur = int(input("Temperatur in Grad Celsius: "))
2 if temperatur <= 0:
3     print("Achtung, Glatteisgefahr!")
4 else:
5     print("Keine Glatteisgefahr.")
```

```
Temperatur in Grad Celsius: 23
Keine Glatteisgefahr.
```

Beschreibung und Ausgaben:

…

4 Die mehrseitige Verzweigung: `if` … `elif` … `else`

Bei mehr als zwei Möglichkeiten gibt es `elif`, kurz für *else if*, „sonst, wenn“.

```
1 note = int(input("Punktzahl (0-100) eingeben: "))
2 if note >= 90:
3     print("Note: sehr gut")
4 elif note >= 75:
5     print("Note: gut")
6 elif note >= 50:
7     print("Note: befriedigend")
8 else:
9     print("Note: nicht bestanden")
```

```
Punktzahl (0-100) eingeben: 78
Note: gut
```

- **Zeile 2:** Ist 80 ≥ 90? Nein.
- **Zeile 4:** Weil die erste Bedingung falsch war, wird die nächste geprüft: Ist 80 ≥ 75? Ja.
- **Zeile 5:** Der zugehörige Block wird ausgeführt.
- **Zeilen 6–9:** Werden übersprungen. Sobald eine Bedingung zutrifft, prüft Python die weiteren **gar nicht mehr**, obwohl auch `80 >= 50` wahr wäre.

Der `else`-Teil fängt alle übrigen Fälle auf. Er darf auch fehlen.

> 📌 **Merksatz: Mehrseitige Verzweigung:** `if … elif … else:` – es wird nur der Block der **ersten** zutreffenden Bedingung ausgeführt. Trifft keine zu, wird der `else`-Block ausgeführt.

⚠️ Stolperstein: die Reihenfolge

🔮 **Vorhersage:** Was gibt das Programm bei 95 Punkten aus?

```
note = int(input("Punktzahl (0-100) eingeben: "))
if note >= 50:
    print("Note: befriedigend")
elif note >= 75:
    print("Note: gut")
elif note >= 90:
    print("Note: sehr gut")
```

```
Punktzahl (0-100) eingeben: 78
Note: befriedigend
```

95 Punkte, aber nur „befriedigend“: `95 >= 50` ist schon wahr, der Rest wird übersprungen. Regel: Bei `>=` die **größte** Grenze zuerst prüfen, bei `<` die **kleinste**.

5 Bedingungen mit `and` und `or`

Eine Bedingung darf alles sein, was `True` oder `False` ergibt, also auch Verknüpfungen aus DS 2.

```
tag = input("Welcher Tag ist heute? ")
if tag == "Samstag" or tag == "Sonntag":
    print("Wochenende! Ausschlafen.")
else:
    print("Wecker stellen.")
```

```
Welcher Tag ist heute? Montag
Wecker stellen.
```

- **Zeile 1:** Hier wird nicht umgewandelt, denn es soll ein Text verglichen werden.
- **Zeile 2:** Wahr, wenn **mindestens einer** der beiden Vergleiche wahr ist.

⚠️ Stolperstein: abgekürztes `or`

🔮 **Vorhersage:** Was gibt das Programm am Montag aus?

```
tag = input("Welcher Tag ist heute? ")
if tag == "Samstag" or "Sonntag":
    print("Wochenende! Ausschlafen.")
```

```
Welcher Tag ist heute? Montag
Wochenende! Ausschlafen.
```

Python liest `"Sonntag"` als eigene Bedingung, und ein nicht leerer Text zählt als wahr. Die Bedingung ist **immer** erfüllt. Schreibe auf **beiden** Seiten von `or` und `and` einen vollständigen Vergleich.

> ⚠️ **Stolperstein:** `if zahl = 0:` meldet `SyntaxError: invalid syntax. Maybe you meant '==' or ':=' instead of '='?` Gemeint ist `==`.

6 Verschachtelte Verzweigungen

Im Block einer Verzweigung darf wieder eine Verzweigung stehen. Nur wenn der Benutzername bekannt ist, wird nach dem Passwort gefragt.

```python
benutzer = input("Benutzername: ")
if benutzer == "alex":
    passwort = input("Passwort: ")
    if passwort == "geheim123":
        print("Willkommen, Alex!")
    else:
        print("Falsches Passwort.")
else:
    print("Unbekannter Benutzer.")
```

```
Benutzername: alex
Passwort: 1234
Falsches Passwort.
```

- **Zeile 2:** Die äußere Verzweigung prüft den Benutzernamen.
- **Zeilen 4–7:** die **innere** Verzweigung. Ihre Blöcke sind acht Leerzeichen eingerückt.
- **Zeile 8:** Das `else` steht genau unter dem äußeren `if` und gehört deshalb dazu.

Ablaufprotokoll für die Eingaben `alex` und `1234` (– heißt: die Variable gibt es noch nicht):

Schritt	Zeile	`benutzer`	`passwort`
1	1	`"alex"`	–
2	2	`"alex"`	–
3	3	`"alex"`	`"1234"`
4	4	`"alex"`	`"1234"`
5	7	`"alex"`	`"1234"`

> 📌 **Merksatz:** Verzweigungen können **verschachtelt** werden. Die Einrückung zeigt, welcher Block zu welcher Bedingung gehört. Ein `else` gehört zu dem `if`, unter dem es genau steht.

📝 Übungen: Schritt für Schritt selbstständig

Das **Musterbeispiel** ist ganz vorgemacht. Bei Übung 2 füllst du Lücken `___`, bei Übung 3 steht nur noch ein Gerüst aus Kommentaren, die Zusatzaufgabe schreibst du allein.

Schritt	Frage
1	**Fälle:** Wie viele Möglichkeiten gibt es? Zwei → `if … else`, mehr → `elif`.
2	**Bedingungen:** Welche Bedingung trennt die Fälle? In welcher Reihenfolge prüfe ich?
3	**Testen:** Welche Eingaben prüfen jeden Fall und jeden **Grenzwert**?

🧩 Musterbeispiel (vorgemacht) – Schwimmbad

Aufgabe: Das Programm fragt nach dem Alter. Unter 18 kostet der Eintritt 3 Euro, sonst 5 Euro.

1. **Fälle:** zwei, also `if … else`.
2. **Bedingung:** `alter < 18`. Ist sie wahr, kostet es 3 Euro.
3. **Testen:** 12 (jugendlich), 30 (erwachsen) und die Grenzwerte 17 und 18.

```python
alter = int(input("Wie alt bist du? "))
if alter < 18:
    print("Eintritt: 3 Euro")
else:
    print("Eintritt: 5 Euro")
```

```
Wie alt bist du? 15
Eintritt: 3 Euro
```

Eingabe	`alter < 18`	Ausgabe
`12`	`True`	`Eintritt: 3 Euro`
`17`	`True`	`Eintritt: 3 Euro`
`18`	`False`	`Eintritt: 5 Euro`
`30`	`False`	`Eintritt: 5 Euro`

📝 Übung 2 – Größte von zwei Zahlen (Lücken füllen)

Das Programm liest zwei Zahlen ein und gibt aus, welche größer ist oder ob beide gleich sind. Das sind drei Fälle. Ersetze jedes `___`.

Kontrolle: 7 und 3 → erste größer, 3 und 7 → zweite größer, 5 und 5 → gleich.

Lückencode

```python
a = float(input("Erste Zahl: "))
b = float(input("Zweite Zahl: "))
if a ___ b:
    print("Die erste Zahl ist größer.")
___ b > a:
    print("Die zweite Zahl ist größer.")
___:
    print("Beide Zahlen sind gleich.")
```

📝 Übung 3 – Altersklassen (mit Gerüst)

Das Programm liest ein Alter ein und ordnet es ein:

- unter 6: „Kleinkind“
- 6 bis unter 13: „Kind“
- 13 bis unter 18: „Jugendlicher“
- ab 18: „Erwachsener“

Ersetze die Kommentare durch Programmzeilen. Teste besonders die **Grenzwerte** 5, 6, 12, 13, 17 und 18.

```python
# Alter als Ganzzahl einlesen
# if: jünger als 6 → Kleinkind
# elif: jünger als 13 → Kind
# elif: jünger als 18 → Jugendlicher
# else → Erwachsener
```

🚀 Zusatzaufgabe – Achterbahn-Check (selbstständig)

Regeln für eine Achterbahn:

- ab 140 cm: Man darf mitfahren.
- 120 cm bis unter 140 cm: nur in Begleitung eines Erwachsenen.
- unter 120 cm: nicht mitfahren.

Frage die Körpergröße (in cm) und ob eine erwachsene Begleitung dabei ist (`ja`/`nein`). Gib die passende Meldung aus. Nutze eine **verschachtelte** Verzweigung.

Kontrolle: 150/nein → mitfahren, 130/ja → in Begleitung, 130/nein → nicht, 110/ja → nicht.

📌 Das Wichtigste in Kürze

- `if Bedingung:` → Block nur bei `True`. Doppelpunkt nicht vergessen!
- Die **Einrückung** (4 Leerzeichen) bestimmt, was zum Block gehört.
- `if … else:` → genau einer der beiden Teile wird ausgeführt.
- `if … elif … else:` → nur die **erste** zutreffende Bedingung zählt. Reihenfolge beachten!
- Bei `or` und `and` auf beiden Seiten einen vollständigen Vergleich schreiben.
- Immer die **Grenzwerte** testen.

Zum Weiterüben: `DS03_Weitere_Uebungen.ipynb`.