

[zurück](#)

Python Teil 1: Variablen, Eingabe, Bedingungen, Fehler abfangen

- **Ziel:**

Dieses Kapitel erklärt die Grundlagen so, dass man sie **nachbauen** und **verstehen** kann – auch ohne Vorkenntnisse. Beispiel-Programme laufen in **Thonny** und **VS Code**.

1. Was ist ein Python-Programm?

Ein Python-Programm ist eine Textdatei mit Endung `.py`, z. B. `01_basics.py`. Python liest die Datei **von oben nach unten** und führt die Befehle aus.



Merke: Reihenfolge ist wichtig.

2. Variablen: Werte speichern

Eine Variable ist wie ein beschrifteter Zettel, auf dem ein Wert steht.

```
name = "Max Mustermann"  
rolle = "Musterschueler"
```

- **Links** steht der **Name der Variable**
- **Rechts** steht der **Wert**
- Das `=` ist eine **Zuweisung** (**kein** mathematisches Gleichheitszeichen)



Tipp:

Variablennamen klein und ohne Leerzeichen, z. B. `name`, `rolle`, `alter`.

3. Ausgabe: Text anzeigen mit `print()`

```
print("Hallo Welt")  
print("Name:", name)
```

`print()` zeigt etwas im Terminal an.

- "Hallo Welt" ist ein Text (String)
- name ist eine Variable

4. Eingabe: Daten abfragen mit `input()`

```
name = input("Name: ")
```

`input()` zeigt eine Frage an und wartet auf Eingabe.



Wichtig:

`input()` liefert **IMMER** Text (String), auch wenn man Zahlen eintippt.

Beispiel:

- Eingabe: 12
- Ergebnis: "12" (Text, keine Zahl)

5. Zahlen aus Text machen: `int()` und `float()`

Wenn wir rechnen oder vergleichen wollen, brauchen wir echte Zahlen.

```
alter_text = input("Alter: ")
alter = int(alter_text)      # ganze Zahl, z. B. 12
```

- `int("12")` → 12
- `float("12.5")` → 12.5

Wenn jemand etwas Nicht-Zahliges eintippt (abc), gibt es einen Fehler. Deshalb machen wir das robust.

6. Fehler abfangen: `try / except`

So verhindern wir Abstürze:

```
while True:
    alter_text = input("Alter (Zahl): ")
    try:
        alter = int(alter_text)
        break
    except ValueError:
        print("Bitte eine gueltige ganze Zahl eingeben, z.B. 42.")
```

Was passiert hier?

- `while True` startet eine Schleife (wiederholt sich)
- `try` versucht den Code auszuführen
- Wenn `int(...)` nicht klappt → `ValueError`
- `except` fängt den Fehler ab und gibt eine Meldung aus
- `break` beendet die Schleife, wenn alles okay ist

**Merke:**

Das ist „professionelles“ Verhalten: Fehler werden kontrolliert behandelt.

7. Entscheidungen treffen: `if / elif / else`

Damit kann Python abhängig von Bedingungen unterschiedliche Dinge tun.

```
if alter < 18:
    print("Du bist minderjaehrig.")
elif alter < 67:
    print("Du bist im Erwerbsalter.")
else:
    print("Du bist im Rentenalter.")
```

- **if** = erste Prüfung
- **elif** = weitere Prüfung (falls `if` nicht zutrifft)
- **else** = sonst

**Wichtig:**

Einrückungen (4 Leerzeichen) sind in Python Pflicht!

8. Unser komplettes Programm (Teil 1)

Datei: 01_basics.py

[01_basics.py](#)

```
# 01_basics.py

import os

# Funktion zum Bildschirm reinigen (clear screen)

def clear_screen():
    # 'nt' steht für Windows, ansonsten Linux/macOS
    os.system('cls' if os.name == 'nt' else 'clear')

# Aufruf der Funktion
clear_screen()

# Variablen initialisieren und Eingabeaufforderung

name = input("Name: ")
rolle = input("Rolle: ")
ort = input("Ort: ")

# Eingabeaufforderung mit Überprüfung und Umwandlung in von String in
Integer

while True:
    alter_text = input("\nAlter (Zahl): ")
    try:
        alter = int(alter_text)
        break
    except ValueError:
        print("Bitte eine gueltige ganze Zahl eingeben, z.B. 42.")

pause = input("\nDruecke Enter zum Fortfahren.")

# Aufruf der Funktion (Bildschirm reinigen)
clear_screen()

# print("\nHallo " + name + " aus " + ort + "!") # Alternative
Ausgabeanweisung

# Ausgabe

print(f"\nHallo {name} aus {ort}!")
print("\nHier sind deine:")
print("\n" + "-" * 20)
```

```
print("ERGEBNISSE")
print("-" * 20)
print("\n--- Profil ---")
print("Name:", name)
print("Rolle:", rolle)
print("Ort:", ort)

print("\n--- Check ---")

# Vergleich (Alterseingabe) und Ausgabe

if alter < 18:
    print("Du bist minderjaehrig.")
elif alter < 67:
    print("Du bist im Erwerbsalter.")
else:
    print("Du bist im Rentenalter.")
pause = input("\nDruecke Enter zum Beenden.")

# Alternative zum Bildschirm reinigen (ANSI Escape Codes)
print("\033[H\033[J", end="")
```

9. Typische Fehler (und wie man sie erkennt)

Fehler 1: Einrückung stimmt nicht

Wenn die Einrückung falsch ist, kommt oft:

IndentationError: unexpected indent

Lösung: 4 Leerzeichen pro Block, keine Mischung aus Tabs/Spaces.

Fehler 2: Zahlenvergleich ohne int()

```
alter = input("Alter: ")
if alter < 18:
    ...
```

Das ist falsch, weil alter ein Text ist. **Lösung:** alter = int(alter)

Fehler 3: Programm endet sofort

Wenn das Terminal sofort schließt, hilft:

```
input("Enter zum Beenden...")
```

10. Mini-Übungen (zum Festigen)

Übung A

Frage zusätzlich Lieblingsfarbe ab und gib sie im Profil aus.

Übung B

Wenn alter kleiner als 0 ist, gib aus: Das Alter kann nicht negativ sein.

Übung C

Gib eine Begrüßung aus, die alle Infos enthält, z. B.: Hallo Max aus Musterstadt, Rolle: Schueler..., Alter: 20

Nächster Baustein (Teil 2 - kommt danach)

Im nächsten Teil lernen wir:

- **Listen**
- **for-Schleifen**
- mehrere Werte speichern und verarbeiten (z. B. mehrere Hosts)

Damit bauen wir dann echte kleine Tools (Netzwerk/Logs/CSV).

From: <http://wiki.nctl.de/dokuwiki/> - Veni. Vidi. sudo rm -rf / vici.

Permanent link: http://wiki.nctl.de/dokuwiki/doku.php?id=python:grundlagen:01_variablen_eingabe_bedingungen_fehler&rev=1772052846

Last update: 25.02.2026 21:54

