Programmiervorkurs 2025

Variablen & Ein-/Ausgabe

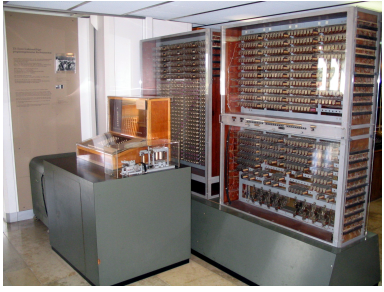
Vorkurs-Schnupsis

29. September 2025

TU Darmstadt

Historisches

Rechenzentrum im Jahre 1917



Z3 von Konrad Zuse

Historisches

Lochkartenleser

Margaret Hamilton und ihr Programm der Apollo 11 Mission

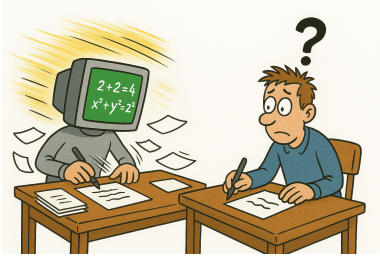
Mit Computern Reden

Wie funktionieren Computer?

Computer ...

- ... sprechen keine natürlichen Sprachen
- ... sind rein-mathematische Maschinen
- ... rechnen Arbeitsschritt für Arbeitsschritt
- ... sind extrem **schnell**

... also wirklich **schnell**



Programme

Was sind Programme?

- Abfolge von Arbeitsschritten in einer fest definierten Schreibweise

  $\Rightarrow$ Analog zu einem Kochrezept
- Die spezifische Schreibweise wird durch die verwendete Programmiersprache festgelegt
- Es gibt unzählige Programmiersprachen

  $\Rightarrow$ Insbesondere gibt es nicht die *eine* Programmiersprache

Python

Die Programmiersprache des Vorkurses

- Python-Programme bestehen aus Text, welcher interpretiert wird

  $\Rightarrow$ Sogenannter Quelltext (engl. Source Code)
- Python-Programme können auf zwei Arten eingegeben werden:
  - Direkt in der Python-Konsole
  - In Dateien, welche an Python übergeben werden

Python Interpreter

Im folgenden werden wir den Python-Interpreter verwenden, um Python-Code auszuführen. Dies wird wie folgt zur Illustration aussehen:

Python-Interpreter

```
> python3
Python 3.13.7 [GCC 15.2.1 20250813] on linux
Type "help" for more information.
>>> print("Hello, World!")
Hello, World!
>>> 1 + 2 * 3
7
```

Live-Coding

Zusammenfassung

Der Mensch schreibt ein Programm als Python-Quelltext

```
movl $0xFF001122, %eax
addl %ecx, %edx
xorl %esi, %esi
pushl %ebx
movl 4(%esp), %ebx
leal (%eax,%ecx,2), %esi
cmpl %eax, %ebx
jnae foo
retl
```

Python übersetzt den Quelltext in Anweisungen für den Computer

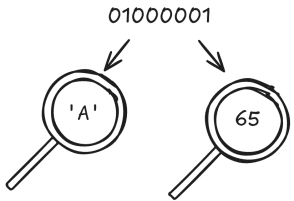
Der Computer führt die Anweisungen aus

Datentypen

Was sind Typen?

Daten werden in einem Computer als Binärzahlen gespeichert. Der Computer selbst kann jedoch nicht wissen, wie diese Zahlen zu interpretieren sind.

- Beispiel: Die Binärzahl `01000001` kann als Zahl (`65`) oder als Zeichen (`'A'`) interpretiert werden.
- Der Typ legt die Interpretation der Daten fest.



Was sind Typen?

- Daten können also unterschiedliche Typen (= Interpretationen) haben
- Die wichtigsten für unseren Vorkurs:

Name	Bedeutung	Beispielwert
`bool`	Wahrheitswert	`True`, `False`
`int`	Ganzzahl	`42`, `-2`
`float`	Fließkommazahl	`21.5`, `-0.135`
`str`	Zeichenkette	`"lorem"`, `'ipsum'`
`NoneType`	Nulltyp	`None`

Operatoren

Mathematische Operatoren in Python

- Python kann mit Zahlen rechnen – ganz einfach:

Operator	Bedeutung	Beispiel
+	Addition	`5 + 2 -> 7`
-	Subtraktion	`5 - 2 -> 3`
*	Multiplikation	`5 * 2 -> 10`
/	Division (immer float)	`5 / 2 -> 2.5`
%	Modulo (Rest)	`5 % 2 -> 1`
//	Ganzzahl-Division (abrunden)	`5 // 2 -> 2`
**	Potenz	`5**2 -> 25`

Mathematische Operatoren

Achtung!: Division mit `/` liefert immer eine Fließkommazahl, auch wenn die Operanden Ganzzahlen sind.

Division

```
>>> 5 / 2
2.5
>>> 5 // 2
2
```

Das ist in anderen Sprachen oft anders! In C, C++, Java, ... ist `5 / 2` immer `2`.

Mathematische Operatoren

- Der abrundende Divisonsoperator `//` ist eher Python-spezifisch und steht in anderen Sprachen üblicherweise nicht zur Verfügung
- Exponentiation: `5**2 = 25`

  $\Rightarrow$ In diesem Vorkurs nicht sonderlich relevant und wird nicht in jeder Sprache so umgesetzt. Meist als Funktion in anderen Spachen (also z.B. `pow(5, 2)`) (Später mehr zu Funktionen an Tag 3)
- Modulo: `5 % 2 = 1`

  $\Rightarrow$ Berechnet den Rest der Division, wird gerne verwendet, um zum Beispiel Teilbarkeit zu prüfen: `10 % 5 == 0`

Zahlen

Grenzen der Darstellung

- In den meisten Sprachen (so auch Python) sind die darstellbaren Zahlen nach oben und unten begrenzt
- Es gibt eine Kurzschreibweise für Zehnerpotenzen: `1.6e6` statt `1600000`
- Fließkommazahlen haben eine begrenzte Genauigkeit und können nicht alles darstellen, klassisches Beispiel:
  `0.1 + 0.2 = 0.30000000000000004`

Vergleichsoperatoren in Python

Neben mathematischen Operationen kann Python auch Vergleichsoperationen, welche verwendet werden, um Bedingungen zu formulieren. Diese liefern immer einen Wahrheitswert (`True` oder `False`) zurück.

Schreibweise	Bedeutung	Beispiel	Wert
`==`	gleich	`3 == 3`	True
`!=`	ungleich	`5 != 5`	False
`>`	größer als	`7 > 7`	False
`>=`	größer gleich	`5 >= 5`	True
`<`	kleiner als	`2 < 5`	True
`<=`	kleiner gleich	`4 <= 4`	True

Live-Coding

Operatorpräzedenz

- Operatoren haben eine festgelegte Reihenfolge, in der sie ausgewertet werden. Diese sind meistens in der Dokumentation der jeweiligen Sprache zu finden.
- Python Dokumentation: Operator Precedence
- Beispiel: `3 + 5 * 2` $\Rightarrow$ 13 (Multiplikation zuerst)
- Beispiel: `(3 + 5)* 2` $\Rightarrow$ 16 (Klammern erzwingen Reihenfolge)
- Beispiel: `5 * 3 > 2 - 1` $\Rightarrow$ True (Multiplikation und Subtraktion vor Vergleich)

Variablen

Variablen

- Variablen sind ein zentrales Konzept in der Programmierung
- Sie ermöglichen es, Daten zu speichern und wiederzuverwenden
- Ohne Variablen wären Programme sehr unpraktisch und schwer lesbar

Was sind Variablen?

- Wie bereits vorher erwähnt kennen Computer nur Bits und diese müssen Interpretiert werden. Genauso arbeitet der Computer auch mit Speicher.
- Variablen sind also Namen für Speicherplätze im Arbeitsspeicher des Computers. Diese dienen dazu, dass wir Menschen uns nicht mit Speicheradressen beschäftigen müssen, wie z.B. `0x7ffeefbff5c8` in Assembly.
- Wir können Variablen erstellen, indem wir ihnen einen Namen geben und einen Wert zuweisen. Diese werden dann automatisch im Speicher abgelegt und bekommen eine Interpretation (Typ) zugewiesen.

Was sind Variablen?

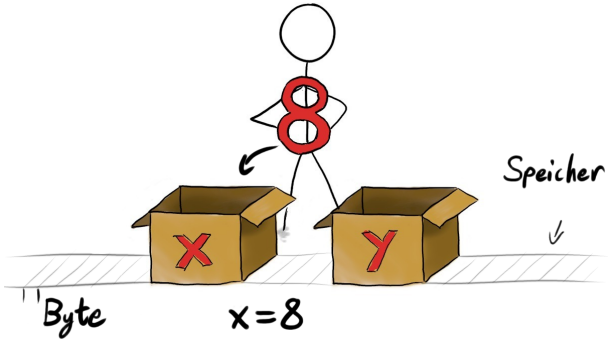


Abbildung: Illustration einer Variable

Namenskonventionen

(In Python)

- Case-sensitive (Groß- und Kleinschreibung wird unterschieden)
- Müssen mit einem Buchstaben oder Unterstrich beginnen
- Dürfen keine Punkte, Bindestriche, Sterne oder andere Sonderzeichen enthalten
- Sollten aussagekräftig sein, einfach nur ”a” und ”b” sind dies meistens nicht
- In Python wird `snake_case` verwendet, nicht `camelCase` oder andere Schreibweisen

Nutzung von Variablen

- Variablen werden in Python mit dem Gleichheitszeichen = erstellt und mit einem Wert befüllt
- Der Wert kann dabei ein beliebiger Ausdruck sein, also auch eine Rechnung oder eine Funktion
- Der Typ der Variable wird automatisch anhand des Wertes bestimmt

```
1 x = 5
2 y = x * 3 + 3
3 y = y * 2
4 print(y)
```

Live-Coding

Zeichenketten

Zeichenketten

- Zeichenketten (engl. Strings) sind eine Abfolge von Zeichen, also Buchstaben, Zahlen und Sonderzeichen
- Hierbei ist bei einer Zahl jedoch nicht die Integer- oder Fließkommazahl gemeint, sondern das Zeichen selbst
- Zeichenketten werden in Python mit einfachen (`'...'`) oder doppelten (`"..."`) Anführungszeichen definiert

  Beispiele: `'Hallo'`, `"12345"`, `'!@#$%'`
- In Python haben die Anführungszeichen keine semantischen Unterschiede

Operatoren

Zeichenketten

```
>>> s1 = 'Lorem'
>>> s2 = "Ipsum"
>>> s1 == s2
False
>>> len(s1)
5
# Mehr hierzu an Tag 3
>>> s1.lower()
'lorem'
>>> s1.upper()
'LOREM'
>>> s1[2]
'r'
```

Operatoren

Konkatenation

- Zeichenketten können mit dem Plus-Operator + zusammengefügt werden
- Dies wird als Konkatenation bezeichnet

  Beispiel: `'Hello, '+ 'World!'` ergibt `'Hello, World!'`
- Wichtig: Zeichenketten können nur mit anderen Zeichenketten verknüpft werden. Es findet keine automatische Konvertierung statt

  `'a'+ 'b'` ist valide, bei `1 + '1'` kommt es zu Fehlern

Live-Coding

Einschub Funktionen

- Hier ein kurzer Einschub zu Funktionen, dass ihr die folgenden Beispiele besser versteht
- Funktionen sind eine benannte Folge von Anweisungen, die eine bestimmte Aufgabe erfüllen
- Weiterhin können Funktionen Parameter entgegennehmen und einen Rückgabewert liefern
- Sie haben die Form `funktionsname(parameter1, parameter2, ...)`
- Funktionen werden später im Vorkurs noch ausführlich behandelt

Konvertierung

- Manchmal ist es notwendig, den Typ eines Wertes zu ändern
- Dies wird als Typumwandlung oder Typkonvertierung bezeichnet
- In Python gibt es eingebaute Funktionen, um Werte in verschiedene Typen zu konvertieren

Funktion	Nutzen	Beispiel
`int()`	Konvertierung zu `int`	`int('3')`
`float()`	Konvertierung zu `float`	`float('3.0')`
`str()`	Konvertierung zu Zeichenkette	`str(3)`
`bool()`	Konvertierung zu `bool`	`bool('True')`

- **Warnung:** Der Datentyp `int` und die Funktion `int()` sind zwei verschiedene Dinge! Selbst wenn diese den selben Namen haben, sind diese semantisch unterschiedlich.

Konvertierung

- Um besser zu verstehen, welche Variablen welchen Typ haben und was die Konvertierungsfunktionen bewirken, können wir die `type()` Funktion verwenden
- Diese Funktion gibt den Typ des übergebenen Wertes zurück

Konvertierung

```
>>> x = 5
>>> type(x)
<class 'int'>
```

Konvertierung

Live-Coding

Ein- und Ausgabe

Ausgabe

- Um Informationen während der Ausführung an den Nutzer auszugeben, gibt es die Funktion `print()`
- Der Funktion wird ein einzelner Parameter übergeben, welcher (bei Bedarf) in eine Zeichenkette umgewandelt wird

```
1 print() # gibt eine leere Zeile aus
2 print(5)
3 print('Lorem ipsum')
```

Ausgabe

Formatierung

- Häufig möchte man bei der Ausgabe nicht nur einen einzelnen Wert ausgeben, sondern zusätzlichen Text oder eine Reihe an Werten.
- Hierfür kann man alle Werte in Zeichenketten umwandeln und zusammenfügen
- Noch besser ist die Nutzung von sogenannten Formatierungsstrings:

```
1 n = 5
2 print('Info: ' + str(n) + ' Dateien heruntergeladen')
3 print(f'Info: {n} Dateien heruntergeladen')
```

Eingabe

- Als Gegenstück zur Ausgabe mit `print()` gibt es auch die Eingabefunktion `input()`.
- Diese gibt eine Zeichenkette zurück, auch wenn nur Ziffern in der Konsole eingegeben wurden
- Optional kann ein String als Parameter übergeben werden, welcher in der Konsole ausgegeben wird: `input('Eingabe: ')`

```
1 x = input()
2 n = input('Bitte gib eine Zahl an: ')
```

Das erste Python Program

Live-Coding

Hilfe

- Python hat eine eingebaute Funktion zum Nachlesen: `help`
- Diese könnt ihr im Interpreter eingeben und dort in der Dokumentation suchen, indem ihr Begriffe eingebt, z.B. ”print”

```
help> print
```

⇒

```
Help on built-in function print in module builtins:

print(*args, sep=' ', end='\n', file=None, flush=False)
    Prints the values to a stream, or to sys.stdout by default.

    sep
      string inserted between values, default a space.
    end
      string appended after the last value, default a newline.
    file
      a file-like object (stream); defaults to the current sys.stdout.
    flush
      whether to forcibly flush the stream.
 Help on print line 1/13 (END) (press h for help or q to quit)
```

Quiz

Quiz

- Was ist ein Datentyp?
- Wie erstellt man eine Variable in Python?
- Nenne drei mathematische Operatoren in Python
- Wie erstellt man eine Zeichenkette in Python?
- Wie findet man den Typ einer Variable heraus?

Ausblick

Ausblick

In der morgigen Vorlesung erwarten uns:

- Verzweigungen und Kontrollfluss
- Ein erster Blick in Schleifen
- Dateioperationen