LaTeX – in all seiner Schönheit

Online Tutorium 1

Florian Sihler ○ 23.04.2020

Die heutigen Themen

1. Die Installation
2. Die Basics
3. Formatierungen
4. Mathe
5. Auflistungen und Tabellen
6. Source Code

Die Basics - Die Installation

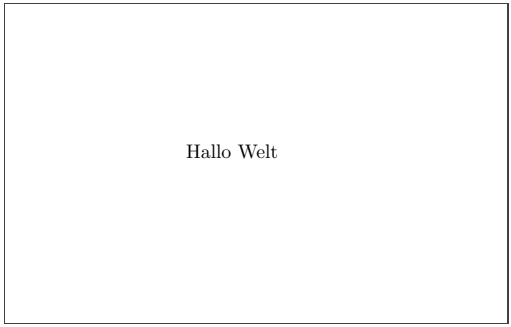
- Keine Installation nötig, es existieren online Compiler wie `https://www.overleaf.com/`
- Auf Linux reicht auf `apt`-basierten Systemen:

```
sudo apt install texlive-full
```

  Sonst: `https://www.tug.org/texlive/tlmgr.html`
- Für Windoof wird MikTeX (`https://miktex.org/download`) in Kombination mit einer IDE wie TeXstudio (`https://www.texstudio.org/#download`) benötigt.
- Für MacOS gibt es eine gute Anleitung: `https://www.latexbuch.de/latex-apple-mac-os-x-installieren/`. Für das iPad bestehen Apps wie TeXpad.

Die Basics - Hallo Welt

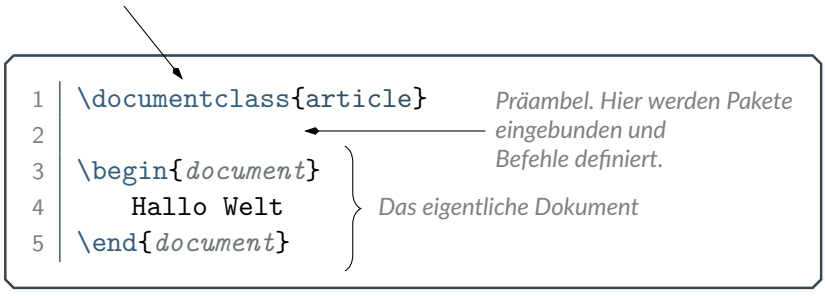
```
1 \documentclass{article}
2
3 \begin{document}
4     Hallo Welt
5 \end{document}
```



- Auf Linux kompilieren mittels: `pdflatex ⟨filename⟩.tex` (Auch für Linux existieren Editoren wie Kile, die viel automatisieren.)
- In Windows/Overleaf: Speichern/klick auf das grüne Dreieck.

Die Basics - Dokumentstruktur

Dokumentklasse, grundlegendes Layout des Dokuments



Die Basics - Ein Paket, was ist das?

- (LaTeX-)Code der von Anderen geschrieben wurde um eine Aufgabe zu bewältigen.
- Werden in der Präambel eingebunden (vor `\begin{document}`).
- Grundlegend: `\usepackage{⟨ Paketname ⟩}`.
- Beispiel: `\usepackage{amsmath}`. (Das Paket liefert einige nützliche Definitionen und Erweiterungen für mathematische Gleichungen.)
- Manche fordern ein Argument, dann lautet die Signatur: `\usepackage[⟨ Argumente ⟩]{⟨ Paketname ⟩}`.
- Beispiel: `\usepackage[utf8]{inputenc}`. (Setzt die (erwartete) Kodierung des Texts auf UTF-8.)
- Was gut ist: für Dokumente in deutscher Sprache reichen in der Regel die immer gleichen Pakete.

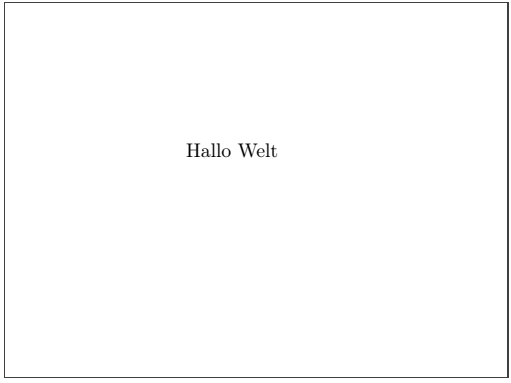
Die Basics - Hallo Welt, jetzt mit Paketen

```latex
\documentclass{article}

\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}

\usepackage[ngerman]{babel}

\begin{document}
    Hallo Welt
\end{document}
```



Nichts anders? Wir werden sehen :D

Die Basics - Wichtige Pakete

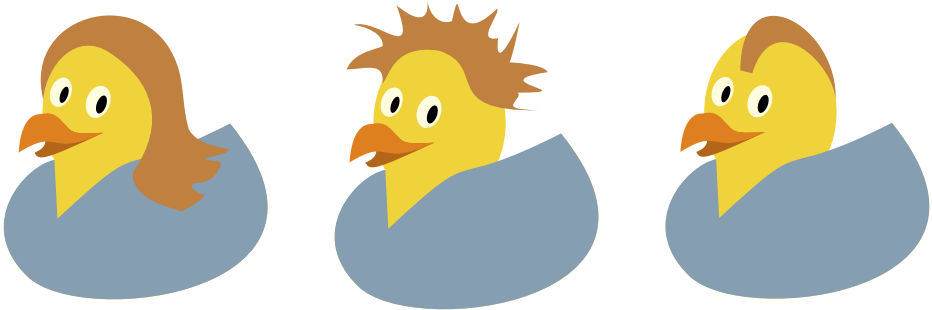
inputenc Lädt die richtige Kodierung für LaTeX

fontenc Kodierung für die Schrift

babel Lädt Worttrennungen und Übersetzungen für die übergebene Sprache

(`ngerman` entspricht *new-german* also der neuen deutschen Rechtschreibung))

- Diese Pakete können eigentlich immer unabhängig vom spezifischen Dokument eingebunden werden. (Die Dokumentationen zu den Paketen erhält man bei einer installierten `texlive-Version` auf Linux/MacOS mittels: `texdoc` ⟨ `Paketname` ⟩. Sonst im Internet „`CTAN` ⟨ `Paketname` ⟩“ suchen.)
- Weitere Pakete folgen ☺.



For*m*atier$\overline{\text{un}_{g}}$en

Formatierungen - Was ist ein Befehl?

- Alle Befehle in LaTeX beginnen mit einem Backslash. (Das Prozentzeichen läutet den Start eines Zeilenkommentars sein. Analog zu `//` in Java.)
- Ein Befehl kann für den Anfang als etwas *magisches* gesehen werden, der zum Beispiel Formatierungsbefehle hinzufügt.
- Die wichtigsten im Textfluss, `\textbf` (**text boldface**), `\textit` (**text italic**) und `\texttt` (**text teletype**):

```
1 \textbf{Hallo Welt},
2 \textit{Hallo Welt},
3 \texttt{Hallo Welt}
```

Hallo Welt, *Hallo Welt*, `Hallo Welt`

- Analog zu einer Methode/Funktion in Java, nehmen Befehle Argumente in geschwungenen Klammern entgegen.

Formatierungen - Ein kunterbunter Farbpalast

- Durch das Paket `xcolor` betreten Farben das Schlachtfeld des Textsatzes.
- Nun kann der Befehl `\textcolor{⟨Farbe⟩}{⟨Text⟩}` verwendet werden um Text farbig zu machen:

```
1 \textcolor{orange}{Hallo Welt},
2 \textcolor{teal}{Hallo Welt}
```

Hallo Welt, Hallo Welt

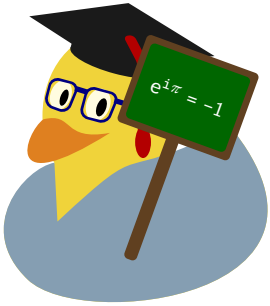
- Ohne Argumente lädt `xcolor` die Farben: *black* ●, *darkgray* ●, *gray* ●, *lightgray* ●, *cyan* ●, *blue* ●, *teal* ●, *purple* ●, *violet* ●, *magenta* ●, *red* ●, *pink* ●, *yellow* ●, *orange* ●, *brown* ●, *olive* ●, *green* ●, *lime* ● und *white* ○.

Formatierungen - Von Para-Grafen und neuen Zeilen

- Neue Zeile im LaTeX-Code wird ignoriert. Zwei oder mehrere Leerzeilen starten einen neuen Paragraphen.
- Beliebig viele Leerfelder (größer Null 😄) kollabieren zu einem.
- Expliziter Start einer neuen Zeile durch `\\` oder `\newline`.
- Expliziter Start eines neuen Paragraphen durch `\par`.
- Beispiel:

```
1 Hallo         Dieter
2 Hallo Welt
3
4 Grüß dich \newline
5 Otto, Ich brauche \\ Beispiele
```

Hallo Dieter Hallo Welt
Grüß dich
Otto, Ich brauche
Beispiele



$$\prod \forall \top \mathbb{H} \varepsilon$$

Mathe - Die (Standard) Umgebungen

- Mathe-Formeln können standardmäßig entweder *inline* oder im *displaystyle* gesetzt werden. (Ersterer versucht die Zeilenhöhe nicht zu verletzen, kann also auch in Fließtext verwendet werden. Der *displaystyle* kann für wichtige Formeln oder längere Rechnungen verwendet werden)
- In jeder Matheumgebung spielen Leerfelder keine Rolle, Abstände werden automatisch „hübsch“ gesetzt.
- „backslash Klammer-auf bis backslash Klammer-zu“ schließt den *inline*-Block ein:

```
1 Hallo 42+3=45,\\
2 ist dies \(42 + 3 = 45\).
```

Hallo 42+3=45,
ist dies $42 + 3 = 45$.

Mathe - Die (Standard) Umgebungen

- Analog eröffnen eckige Klammern eine Matheumgebung im *displaystyle*:

```
1 Hallo \[42 + 3 = 45\]
```

Hallo

$$42 + 3 = 45$$

(Beachte den Start der neuen Zeile, die übrigens zentriert wird. Manche Ausdrücke werden im *displaystyle* großzügiger gesetzt.)

- Sehr hilfreich ist das bereits erwähnte Paket `amsmath`. Es fügt weitere *Umgebungen* hinzu.
- Neben `amsmath` existiert auch `mathtools` (bindet `amsmath` intern mit ein)

Mathe - Grundlegende Befehle

- Viele Operatoren sind direkt über die Tastatur erreichbar (`+`, `-`, `*`, `<`, `>`) und können problemlos verwendet werden.
- Griechische Buchstaben: `\(\alpha\)` (α), `\(\beta\)` (β) und `\(\gamma\)` (γ), … sowie `\(\Gamma\)` (Γ), `\(\Omega\)` (Ω) und `\(\Pi\)` (Π), …
- Operatoren wie `\(\sin\)` (sin), `\(\cos\)` (cos) und `\(\tan\)` (tan), `\(\exp\)` (exp) oder `\(\lim\)` (lim), …
- Auch nützlich: `\(\cdot\)` ($\cdot$), `\(\pm\)` ($\pm$), `\(\sum\)` ($\sum$), `\(\prod\)` ($\prod$), `\(\to\)` ($\to$), `\(\infty\)` (∞), `\(\int\)` ($\int$), `\(\neq\)` ($\neq$), `\(\mapsto\)` ($\mapsto$), `\(\leq\)` ($\leq$) und `\(\geq\)` ($\geq$).

Mathe - Exponent und Index

- „Durch das Hütchen setzt man Text nach oben, durch einen Unterstrich nach unten“
- Beispiel:

```
\(x^{2}_{3}\) \(y^{x_i}_{Hi}\) \\
\(\lim_{x \to \infty} x^{2^{2}}\)
```

x^{2}_{3} $y^{x_i}_{Hi}$
$\lim_{x \to \infty} x^{2^{2}}$

- Bei einzelnen Zeichen kann die Klammer weggelassen werden.
- *Übrigens: Ist `amsmath` eingebunden, kann mittels `\text` Text im Mathemodus geschrieben werden*:

```
\(Hallo Welt \text{ Hallo Welt}\)
```

HalloWelt Hallo Welt

Mathe - Bruch, Wurzel, Binomiales

- Bruch durch `\frac{⟨Zähler⟩}{⟨Nenner⟩}`
- Binomialkoeffizient mit `\binom{⟨n⟩}{⟨k⟩}`
- Wurzel durch `\sqrt{⟨Mathe⟩}`
- Beispiel:

```
1 \(\frac{n!}{k!(n-k)!} = \binom{n}{k}\) \\
2 \(\sum_{i = 1}^n i =
3     \frac{n(n-1)}{2} \cdot \sqrt{1}\)
```

$$\frac{n!}{k!(n-k)!} = \binom{n}{k}$$
$$\sum_{i=1}^{n} i = \frac{n(n-1)}{2} \cdot \sqrt{1}$$

- Kann beliebig verschachtelt werden.

Mathe - Automatische Größe

- Die Größe einiger Klammern und „Mathe-Symbole“ kann mittels `\left`, `\middle` und `\right` automatisch bestimmt werden.
- *Wichtig:* Ein `\left` benötigt immer `\right`.
- Beispiel:

```
1 \([\frac{-13}{5},\infty)\),
2 \(\left[\frac{-13}{5},\infty \right)\)
```

$[\frac{-13}{5}, \infty)$, $\left[\frac{-13}{5}, \infty\right)$

- Wenn nur eine Klammer gewünscht kann ein Punkt platziert werden:

```
1 \(\left[\frac{-13}{5},\infty \right.\)
```

$\left[\frac{-13}{5}, \infty\right.$

Mathe - Mengen

- Eine Menge kann mittels `\mathbb{⟨Buchstabe⟩}` definiert werden:

```
1 \(\mathbb{N}, \mathbb{Q}, \mathbb{R}\)
```

$\mathbb{N}, \mathbb{Q}, \mathbb{R}$

- Für Mengen-Operationen gibt es: `\(\in\)` ($\in$), `\(\notin\)` ($\notin$), `\(\subset\)` ($\subset$), `\(\supset\)` ($\supset$), `\(\subseteq\)` ($\subseteq$), `\(\supseteq\)` ($\supseteq$), `\(\setminus\)` ($\setminus$), `\(\times\)` ($\times$), `\(\emptyset\)` ($\emptyset$) und `\(\varnothing\)` ($\varnothing$).
- Beispiel:

```
1 Sei \(x \in \mathbb{N}\) sowie
2 \(M \subseteq \mathbb{R}^+\).
```

Sei x $\in \mathbb{N}$ sowie M $\subseteq \mathbb{R}^+$.

- komplexe Zahlen: `\(\Re\)` ($\Re$) und `\(\Im\)` ($\Im$).

Mathe - Logik

- Junktor: \(\neg\) (¬), \(\implies\) (⟹), \(\iff\) (⟺), \(\land\) (∧) und \(\lor\) (∨).
- Alternativen: \(\leftarrow\) (←), \(\Leftarrow\) (⇐), \(\rightarrow\) (→), \(\Rightarrow\) (⇒), \(\leftrightarrow\) (↔), \(\leftrightarrow\) (↔) und \(\nearrow\) (↗), ...
- Quantoren: \(\exists\) (∃), \(\nexists\) (∄) und \(\forall\) (∀).
- Beispiel:

```
1 \(\forall h \in \text{Hamster}
2   \exists m \in \text{Menschen}:
3   \text{mag}(m,h) \)
```

$$\forall h \in \text{Hamster} \exists m \in \text{Menschen} : \text{mag}(m, h)$$

Mathe - Abstände

- In Mathe: `\,` sowie `\:` und `\;` für Abstände. (`\!` erzeugt einen negativen Abstand.)
- Beispiel:

```
\(\forall h \in \text{Hamster}\:
  \exists m \in \text{Menschen}
  \,: \text{mag}(m,h) \)
```

$\forall h \in \text{Hamster}\ \exists m \in \text{Menschen}\ : \text{mag}(m, h)$

Mathe - Matrizen

- Matrix mittels *pmatrix* (*matrix* ist ohne Klammer).
- Ausrichten durch Und, neue Zeile durch \\.
- Beispiel:

```
\(\begin{pmatrix}
  1 & 2 & 3 & 42 \\
  219 & \cdots & 1 & 2 \\
  x_1 & 3 & 3 & 7
\end{pmatrix}\)
```

$$\begin{pmatrix} 1 & 2 & 3 & 42 \\ 219 & \cdots & 1 & 2 \\ x_1 & 3 & 3 & 7 \end{pmatrix}$$

Mathe - Fallunterscheidungen

- Partiell definierte Funktionen können ähnlich zu *matrix* mittels *cases* realisiert werden.
- Beispiel:

```
1 \(\min(x,y) = \begin{cases}
2   x, & \text{ falls } x < y, \\
3   y, & \text{ sonst.}
4 \end{cases}\)
```

$$\min(x, y) = \begin{cases} x, & \text{falls } x < y, \\ y, & \text{sonst.} \end{cases}$$

- Doch was ist dieses *cases* überhaupt?

Exkurs: Umgebungen

- An sich bereits bereits bekannt: *document*.
- Wird eingeleitet durch `\begin{⟨Name⟩}` und beendet durch `\end{⟨Name⟩}`.
- Innerhalb einer Umgebung können Befehle eine andere Bedeutung haben, oder sogar einzelne Zeichen eine besondere Rolle einnehmen.
- Vorgriff: Tabelle (Umgebung: *tabular*), hier trennt das Und-Symbol die Spalten und `\\` die Zeilen:

```
\begin{tabular}{l|c|r}
  Hallo & Welt & Na du? \\
  Das & ist \\
  doch & echt & toll!
\end{tabular}
```

Hallo	Welt	Na du?
Das	ist	
doch	echt	toll!

Mathe - align und align*

- So liefert `amsmath` die Umgebung *align*. Hier kann das Und-Symbol verwendet werden um Gleichungen horizontal auszurichten. Wieder startet `\\` eine neue Zeile:

```
\begin{align}
\lim_{n \to \infty} 3^2 \cdot \frac{1}{n} &= \lim_{n \to \infty} 9 ↩
    \cdot \lim_{n \to \infty} \frac{1}{n} \\
    &= 9 \cdot 0 = 0
\end{align}
```

$$\lim_{n\to\infty} 3^2 \cdot \frac{1}{n} = \lim_{n\to\infty} 9 \cdot \lim_{n\to\infty} \frac{1}{n} \tag{1}$$

$$= 9 \cdot 0 = 0 \tag{2}$$

Mathe - align und align*

- Analog funktioniert `align*`, allerdings werden hier die Gleichungen nicht durchnummeriert. (Es sind beliebig viele Zeilen möglich.)
- Wenn diese Umgebungen über mehrere Seiten umbrechen dürfen, setze: `\allowdisplaybreaks[2]` in der Präambel.

Mathe - Die wichtigsten Befehle

- Es existieren viel zu viele, die man auch nicht auswendig lernen muss (das meiste kommt mit der Zeit intuitiv).
- Beispiel Gleichung 1:

```
\[
    \sum_{i = 0}^{42} \prod_{j = 0}^i \lim_{x_i^2 \to x_j^{42}}
    = \frac{3 \cdot \frac{\pi}{2} +
        \left( 1 + \frac{\Omega_\theta}{2}\right)}
        {\sqrt{13} \pm \sin(\lceil x \cdot \tau \rceil)}
\]
```

$$\sum_{i=0}^{42} \prod_{j=0}^{i} \lim_{x_i^2 \to x_j^{42}} = \frac{3 \cdot \frac{\pi}{2} + \left(1 + \frac{\Omega_\theta}{2}\right)}{\sqrt{13} \pm \sin(\lceil x \cdot \tau \rceil)}$$

Mathe - Die wichtigsten Befehle

- Beispiel Gleichung 2:

```
\[
    \forall i \in X = \mathbb{N} \exists x_i, x_j \in D
        = \mathbb{R} : (x_i \leq 42 \land x_j \neq -x_i)
        \lor D \cap X = D \cup X = \emptyset \text{ so nämlich.}
\]
```

$$\forall i \in X = \mathbb{N} \exists x_i, x_j \in D = \mathbb{R} : (x_i \leq 42 \land x_j \neq -x_i) \lor D \cap X = D \cup X = \emptyset \text{ so nämlich.}$$

Mathe - Auch nützlich

- Für komplexere Gleichungen: *gather*, *gather**, *alignat*, *alignat**, *array* und *equation*.
- Und Vieles mehr 😄.
- In der Regel: „Ich brauche X, ich befrage das Internet“
- Wenn man nur das Aussehen kennt:
  `http://detexify.kirelabs.org/classify.html` (Da darf man malen ☺.)

Zitternd greift er die Feder,

Mühsam das letzte Blatt.

Legt es auf den Einband aus Leder,

Mit dem alles begonnen hat.

Er beginnt sein Werk zu vollenden,

An diesem schönen Sommertag.

Widmet die Worte den Enten

Die er eben noch gesehen hat.

Auflistungen und Tabellen

(Nein, nicht alles handelt von Enten.)

Auflistungen - Die wichtigsten

Latex liefert drei wichtige Umgebungen mit: *itemize*, *enumerate* und *description*.

In ihnen kann mithilfe von `\item` ein neuer (Unter-)Punkt gesetzt werden.

Häufiger Fehler: Die Umgebungen benötigen mindestens ein `\item`, sonst wirft Latex den Fehler:

```
! LaTeX Error: Something's wrong--perhaps a missing \item.
```

`\item` kann mittels eckiger Klammern ein Argument übergeben bekommen! (Dies ist wichtig für *description*).

Auflistungen - Beispiele

- Beispiel mit *itemize*:

```
\begin{itemize}
    \item Hallo
    \item Welt
    \item Hihi
\end{itemize}
```

- Hallo
- Welt
- Hihi

- Übriges: Diese Umgebungen lassen sich bis zu einer gewissen Tiefe (in der Regel vier) verschachteln!

Auflistungen - Beispiele

- Beispiel mit *enumerate*:

```
\begin{enumerate}
    \item Hallo
    \item Welt
    \item Hihi
    \item Beachte: \begin{enumerate}
        \item Na du?
        \item Enteeeen
    \end{enumerate}
\end{enumerate}
```

1. Hallo
2. Welt
3. Hihi
4. Beachte:
    4.1 Na du?
    4.2 Enteeeen

Auflistungen - Beispiele

- Beispiel mit *description*:

```
\begin{description}
    \item[Wichtig:] Hallo
    \item[Echt wichtig:] Welt
    \item[Steinzeit:] Hihi
\end{description}
```

Wichtig: Hallo
Echt wichtig: Welt
Steinzeit: Hihi

- Das Aussehen kann mittels `enumitem` quasi beliebig gestaltet werden.
- Zeilen- sowie Seitenumbrüche werden automatisch mit Einschub vollzogen.

Auflistungen - Description im Dokument

- *description* innerhalb eines „normalen“ Dokuments:

```
1 \documentclass{article}
2
3 \begin{document}
4     \begin{description}
5         \item[Wichtig:] Hallo
6         \item[Echt wichtig:] Welt
7         \item[Steinzeit:] Hihi
8     \end{description}
9 \end{document}
```

Wichtig: Hallo

Echt wichtig: Welt

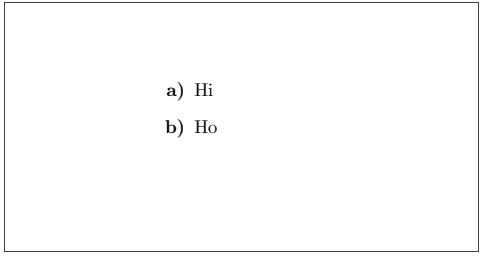
Steinzeit: Hihi

Auflistungen - Bonus: Aufgabennumerierung

- Mittels `enumitem` können wir eine neue Umgebung erzeugen, die gewünscht nummeriert:

```latex
\documentclass{article}
\usepackage{enumitem}
\newlist{aufgaben}{enumerate}%
                  {1}
\setlist[aufgaben]%
    {label=\textbf{\alph*)}}

\begin{document}
    \begin{aufgaben}
        \item Hi
        \item Ho
    \end{aufgaben}
\end{document}
```



Auflistungen - Bonus: `enumitem`

- Das Setzen von `label` (bei `\setlist`) erlaubt anstelle von `\alph*` andere „Einstellungen“ für den Iterator: `\alph*`, `\Alph*`, `\arabic*`, `\Roman*` und `\roman*`.
  (Hierfür und für andere Einstellungen: Dokumentation lesen/Internet fragen/Rum-Probieren/…!)
- Diese Optionen können auch nur einmalig angewandt werden:

```
1 \documentclass{article}
2 \usepackage{enumitem}
3 \begin{document}
4     \begin{enumerate}[label=↩
          \textbf{\Roman*)}]
5         \item Hi
6         \item Ho
7     \end{enumerate}
8 \end{document}
```

I) Hi

II) Ho

Tabellen - Vo(m/n) Spalten

- Wichtige Umgebung: *tabular*
- Benötigt als Argument die Ausrichtungen der jeweiligen Spalten: `l` - linksbündig, `r` - rechtsbündig, `c` - zentriert und `p{# cm}` - linksbündig, # cm breit.
- Innerhalb der Tabelle trennt das Und-Symbol (&) die Spalten und `\\` die Zeilen.
- Beispiel:

```
1 \begin{tabular}{lrcp{0.75cm}}
2   Hihi & Huhu & Hoho & Hallo Du \\
3   A & B & C & D
4 \end{tabular}
```

Hihi	Huhu	Hoho	Hallo Du
A	B	C	D

- die Länge der `p`-Spalte kann auch in Millimeter oder ähnlichem angegeben werden.

Tabellen - Linien

- Bei der Spaltenangabe kann die Pipe (|) für einen vertikalen Strich verwendet werden. In der Tabelle selbst setzt `\hline` eine horizontale Linie. (Stilhinweis: Vertikale Linien sind grauenvoll und sollten vermieden werden ☺.)
- Für detailliertere Spezifikation der Linien: Paket `hhline`.
- Beispiel:

```
\begin{tabular}{l||r|c|p{0.75cm}|}
  \hline
  Hihi & Huhu & Hoho & Hallo Du \\
  \hline \hline
  A & B & C & D \\ \hline
\end{tabular}
```

Hihi	Huhu	Hoho	Hallo Du
A	B	C	D

Tabellen - In Hübsch

- Das Paket `booktabs` liefert: `\toprule`, `\midrule` und `\bottomrule` für `\hline`.
- Verwendung wie bisher:

```
\begin{tabular}{lrcp{0.75cm}}
  \toprule
  Hihi & Huhu & Hoho & Hallo Du \\
  \midrule
  A & B & C & D \\ \bottomrule
\end{tabular}
```

Hihi	Huhu	Hoho	Hallo Du
A	B	C	D

- Booktabs beißt sich „absichtlich“ mit vertikalen Linien 😉.

Tabellen - Mathe-Spalte

- Hinweis: Wenn komplette Tabelle als Mathe-Tabelle empfiehlt sich: *array*.
- Sonst kann jede Spalte mit einem gewissen Code begonnen und beendet werden!
- Hierzu wird die Syntax: `>{⟨pre⟩}⟨coltype⟩<{⟨post⟩}` verwendet
- Damit geht:

```
\begin{tabular}{>{\(}l<{\)}l}
  Mathe & Kein Mathe \\
  42^3 + x_i & immer noch nö
\end{tabular}
```

Mathe	Kein Mathe
$42^3 + x_i$	immer noch nö

Tabellen - Super-Spalten

- Dies ermöglicht viel Freude 😄.
- Beispiel:

```
\begin{tabular}%
  {>{\(}l<{\)}>{\itshape}lr<{Hey}}
  Mathe & kursiv & Stups: \\
  42^3 & hi & Wups:
\end{tabular}
```

Mathe	*kursiv*	Stups:Hey
42^3	*hi*	Wups:Hey

Tabellen - Eigene Spalten

- Schreiben wieder redundant ☹. Paket: `tabularx`.
- Mittels `\newcolumntype` lassen sich eigene Spalten definieren:

```
\newcolumntype{u}{>{\(}c<{\)}}
\begin{tabular}{ul}
  Mathe & Kein Mathe \\
  42^3 + x_i & immer noch nö
\end{tabular}
```

Mathe	Kein Mathe
$42^3 + x_i$	immer noch nö

- Liefert zudem: Die `X`-Spalte, vergleichbar zu `p`, allerdings wird Breite automatisch bestimmt - funktioniert nur innerhalb von `tabularx`.

Tabellen - Lohnt sich anzuschauen

- Die Umgebung *table*. (Beschriften und Auflisten von Tabellen.)
- Das Paket `longtable`. (Wenn die Tabelle über mehrere Seiten gehen soll/Seitenumbrüche beinhaltet)
- Das Paket `ltablex`. (Macht `tabularx` effektiv zu einem `longtable`)
- Das Paket `multirow`. (Vereinigen von mehreren Spalten/Zeilen.)
- Das Paket `tabu`. *Warnung: tabu ist aktuell „unmaintained“ und sollte deswegen mit Vorsicht genossen werden!. Es liefert trotzdem schöne Ideen* ☺. (Ermöglicht gezieltes Einfärben, Spaltenkontrolle, …)

Listings und Pseudocode

Listings - Motivation

Wie kann so eine adrette Ente erschaffen werden? So: (Der komplette Aufwand betrug: 23 Minuten)

```
% ...
\begin{tikzpicture}[scale=1.75,every node/.style={transform shape}]
    \foreach \layer/\scalefactor/\stepfirst/\ifirst in {-.75/0.25/-1.52/-1.56,-.25/0.25/-1.39/-1.5,0/0.6/-1.3/-1.5}{
        \foreach \col in {\ifirst,\stepfirst,...,3.25}{
            \pgfmathsetmacro{\dodraw}{int(3.9*abs(rand))}
            \pgfmathsetmacro{\xoff}{0.25*(1+\stepfirst)*rand}
            \pgfmathsetmacro{\dopacity}{0.4+\scalefactor}
            \ifnum\dodraw>0
            \foreach[count=\linegrad] \line in {3,2.85,...,-0.5}{
                \pgfmathsetmacro{\yoff}{0.125*(1+\stepfirst)*rand}
                \pgfmathsetmacro{\rndsymb}{int(int(8*abs(rand))+1)}
                \pgfmathsetmacro{\shading}{(100-15*\linegrad)/100}
                \pgfmathsetmacro{\doydraw}{int(int(30-\linegrad)*abs(rand))}
                \ifnum\doydraw>1
                    \ifnum\linegrad=24\relax% last
                        \def\nodetarcol{matrixgreenhl} \else
                        \def\nodetarcol{matrixgreen} %
                    \fi
                \else\def\nodetarcol{matrixgreenhl}\breakforeach\fi%force last
                \node[\nodetarcol,opacity=\dopacity] at(\col+\xoff,\line-\yoff,\layer) {\scalebox{\scalefactor}{\RndChinese{\rndsymb}}};
            }\fi
        }
    }
    \duck[sunglasses=black,jacket=black,parting=black]
    \node[below] at(1,-0.5) {\sffamily Listings und Pseudocode};
\end{tikzpicture}
% ...
```

Listings - Es gibt Alternativen

- Ich verwende `listings`, es existiert aber auch (zum Beispiel) `minted`. (`minted` verwendet die `pygments`-Bibliothek (Python), fordert also die Installation weiterer Programme und *shell escape*!)
- Das Einbinden des Pakets stellt die Umgebung *`lstlisting`* zur Verfügung:

```
1 | \begin{lstlisting}[language=java]
2 | public static void
3 |         main(String[] args){
4 |     System.out.println("Hi");
5 | }
6 | \end{lstlisting}
```

```
public static void
        main(String[] args){
    System.out.println("Hi");
}
```

- Sowie `\lstinline` und `\lstinputlisting`. (Hierfür Dokumentation oder Erklärdokument betrachten.)

Listings - Eigener Stil

- Hinweis:
  `https://gist.github.com/EagleoutIce/1490bfd4d71eef73b032670921eab69a`
- Ermöglicht mittels `\lstdefinestyle` Konfiguration der Umgebung. Beispiel:

```
\lstdefinestyle{MeinStil}{
    breaklines     = true,
    stringstyle    = \color{teal},
    keywordstyle   = \color{orange},
    basicstyle     = \ttfamily,
    commentstyle   = \color{gray}\itshape,
    numbers        = left
}
```

Listings - Eigener Stil

- Ermöglicht:

```
\begin{lstlisting}[language=java,style=↩
      MeinStil]
public static void
        main(String[] args){
    System.out.println("Hi");
}
\end{lstlisting}
```

```
public static void
        main(String[] args){
    System.out.println("Hi");
}
```

- Die explizite Angabe für jedes `lstlisting` kann mittels:

  `\lstset{language=java,style=MeinStil}`

  erspart werden.

Listings - Eigene Sprache

- Um zum Beispiel eigene Keywords definieren zu können:

```
\lstdefinelanguage{MeinJava}{
    language=java,
    alsoletter={@_},
    comment=[l]{//},
    morecomment=[s]{/*}{*/},
    keywordsprefix={@},
    morekeywords={String, var},
    morekeywords=[2]{System}
}
```

Listings - Eigener Stil

- Ermöglicht:

```
\lstset{%
    language=MeinJava,style=MeinStil}
\begin{lstlisting}
public static void
        main(String[] args){
    System.out.println("Hi");
}
\end{lstlisting}
```

```
public static void
        main(String[] args){
    System.out.println("Hi");
}
```

- Einziges verbleibendes Problem: Umlaute erzeugen einen Fehler!

Listings - Mähr Zöichän

- Wir verwenden *literates*. (Ersetzungsregeln für Zeichen(-ketten).)
- Definieren (Präambel):

```
\lstset{literate={ö}{{\"o}}1 {ü}{{\"u}}1 {Ä}{{\"A}}1
       {ä}{{\"a}}1 {Ö}{{\"O}}1 {Ü}{{\"U}}1 {ß}{{\ss}}1
       {:hello:}{{Hey du}}5
}
```

- Die Zahl steht für die Anzahl der ersetzten Zeichen. (So bedeutet `{:hello:}{{Hey du}}5`: ersetze „`:hello:`“ mit „`Hey du`“.)

Listings - Mähr Zöichän

- Ermöglicht:

```
\lstset{%
    language=MeinJava,style=MeinStil}
\begin{lstlisting}
public static void
        main(String[] args){
    System.out.println(":hello: dü dää")↩
        ;
}
\end{lstlisting}
```

```
public static void
        main(String[] args){
    System.out.println("Hey␣du␣dü↩
        ␣dää");
}
```

- Es empfiehlt sich die Literates aus `https://gist.github.com/EagleoutIce/1490bfd4d71eef73b032670921eab69a` einfach zu kopieren und gegebenenfalls zu erweitern.

Pseudocode - Das Paket der Wahl

- Ich verwende hier `algorithm2e`. Das Ergebnis kann optisch (leicht) abweichen (einfach Dokumentation anschaun')
- Wir erhalten (unter anderem) die Umgebung *algorithm*.
- Beispiel:

```
1 \begin{algorithm}
2     hi
3 \end{algorithm}
```

```
1 hi
```

Pseudocode - Benennung, In- und Output

- Benennen mittels `\caption`.
- Input- und Ergebnis-Felder mittels: `\KwIn` und `\KwOut`
- Beispiel:

```
\begin{algorithm}
    \KwIn{Eine Ente}
    \KwOut{Einer Super-Ente}
    hi
    \caption{Super Enten-Transformator}
\end{algorithm}
```

Input: Eine Ente
Output: Einer Super-Ente
1 hi
Algorithmus 1: Super Enten-Transformator

- Analog: `\KwData` und `\KwResult`.

Pseudocode - Schleifen

- Schleife mit `\For` oder `\While`.
- Beispiel:

```
1 | \begin{algorithm}
2 |     \For{i = 0 \KwTo 42}{
3 |         hi
4 |     }
5 | \end{algorithm}
```

1 **for** *i = 0* **to** *42* **do**
2 | hi
3 **end**

```
1 | \begin{algorithm}
2 |     \While{Ente gut}{
3 |         Alles gut.
4 |     }
5 | \end{algorithm}
```

1 **while** *Ente gut* **do**
2 | Alles gut.
3 **end**

Pseudocode - Fallunterscheidungen

- Schleife mit `\If` und `\Else` (Kurz: `\eIf`) oder `\Switch` und `\Case`.
- Beispiel:

```
\begin{algorithm}
    \eIf{Ente gut}{
        Alles Gut
    }{Oh Goooott}
\end{algorithm}
```

if *Ente gut* **then**
    Alles Gut
else
    Oh Goooott
end

- Zeilenende mit: `\;`:

```
\begin{algorithm}
\eIf{Ente gut}{Alles Gut\;}
    {Oh Goooott\;}
\end{algorithm}
```

if *Ente gut* **then**
    Alles Gut;
else
    Oh Goooott;
end

Pseudocode - Kommentare und Kleinigkeiten

- Wenn kein Semikolon ausgegeben werden soll (bei `\;`) einfach: `\DontPrintSemicolon` zu Anfang des Codes.
- Kommentare können mittels `\tcc` (C-Style) oder `\tcp` (C++-Style) gesetzt werden.
- Beispiel:

```
\begin{algorithm}
    \tcc{Enten maaarsch:}
    \eIf{Ente gut}{
        Alles Gut\;
    }{Oh Goooott\;}
    \tcp{Tapp, tapp, tapp}
\end{algorithm}
```

```
/* Enten maaarsch: */
if Ente gut then
    Alles Gut;
else
    Oh Goooott;
end
// Tapp, tapp, tapp
```

- Siehe Dokumentation für weitere Feinheiten.