Übungsblatt:	13	**1. Abgabepartner*in:**	Ada Lovelace
Aufgabe:	37	**2. Abgabepartner*in:**	Donald Knuth
Abgabegruppe:	42	**3. Abgabepartner*in:**	Alan Turing

Dies ist eine LaTeX-Vorlage für Übungszettelabgaben im Fach Informatik, die das `minted`-Paket nutzt. Im Gegensatz zu den standardmäßigen `listings`-Umgebungen, für die in meinem Git-Repository[1] ebenfalls eine Vorlage bereitgestellt wird, muss man einige Vorbereitungen treffen, damit `minted` (und auch diese Vorlage) benutzt werden kann. Belohnt wird man dafür mit einem schöneren und deutlich differenzierteren automatischen Syntax Highlighting. Nachfolgend gibt es ein paar Hinweise, wie diese Vorlage zu verwenden ist.

Disclaimer

Ich kann leider grundsätzlich keinen LaTeX-Support anbieten und verweise daher auf gängige Suchmaschinen und die TeX-Community von StackExchange[2]. Sucht man den LaTeX-Befehl für ein bestimmtes Symbol, ist Detexify[3] praktisch.

Verwendung

`minted` basiert auf der Programmiersprache Python[4]. Damit man es zum Laufen bekommt, muss man allerdings keine Kenntnisse diesbezüglich mitbringen, sondern lediglich eine aktuelle Version installieren. Im Anschluss muss noch `Pygments` eingerichtet werden. Details dazu finden sich in Abschnitt 2.1 der `minted`-Dokumentation[5].

Als Test sollte man versuchen, die Datei `abgaben-minted.tex` in einem TeX-Editor zu öffnen und zu kompilieren. Dabei handelt es sich um den TeX-Code zu genau diesem Dokument. Wenn das nicht klappt, kann das folgende Gründe haben:

- In den Umgebungsvariablen des Systems fehlt das Verzeichnis der LaTeX-Distribution oder zu Python.
- Der Compiler wird nicht mit der Shell-Escape-Option aufgerufen. Ggfs. sollte in den Einstellungen des Editors nachgeschaut werden, dass `pdflatex` mit der Option `-shell-escape` aufgerufen wird.
- Irgendwas anderes. Man prüfe die Ausgabe des Compilers und ziehe ggfs. die `minted`-Dokumentation zurate.

Wenn alles geklappt hat, kann man diese Vorlage verwenden. Ich habe als Einstiegshilfe unten verschiedene Beispiele – unter anderem zum Einbinden von Quellcode – angehängt. Die Verwendung der entsprechenden TeX-Befehle sollte anhand der Datei `abgaben-minted.tex` erschließbar sein. Weitere Hinweise und Konfigurationsmöglichkeiten findet man in der Paketdokumentation.

Nicht vergessen: Ein automatisches Syntax Highlighting entbindet **niemals** von der Pflicht, gut lesbaren und gut nachvollziehbaren Code zu produzieren.

Phil Steinhorst
`https://github.com/phist91/latex-templates`

[1] `https://gitlab.com/phist91/latex-templates`
[2] `https://tex.stackexchange.org/`
[3] `http://detexify.kirelabs.org/`
[4] `https://www.python.org/downloads/`
[5] `https://ctan.org/pkg/minted?lang=de`

Beispiele

Textformatierungen

Eine Auflistung:

- Dieser Text ist **fett**.
- Dieser Text ist *kursiv*.
- Dieser Text ist unterstrichen.
- Dieser Text ist rot.
- Dieser Text ist anklickbar.

Eine Aufzählung:

(a) Eine Liste,

(b) aber mit

(c) Nummerierung!

Eine zentrierte Tabelle:

Etwas	gewöhnungsbedürftig,
aber vielseitig und machbar!	

Die gleiche Tabelle, aber mit Beschreibung:

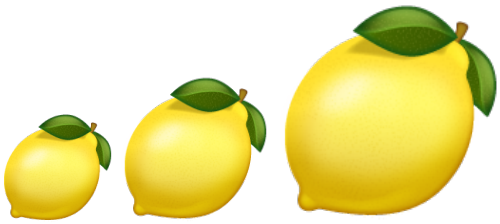
Etwas	gewöhnungsbedürftig,
aber vielseitig und machbar!	

Tabelle 1: Viele Editoren bieten Assistenten zum Anlegen von Tabellen wie dieser.

Grafiken

Eine eingebundene Grafikdatei:

Die gleiche Grafik, in verschiedenen Größen:



(a) klein (b) mittel (c) groß

Abbildung 1: Zitronen in verschiedenen Größen.

Eine TikZ-Grafik mit Beschreibung:

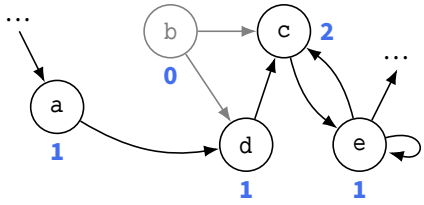


Abbildung 2: TikZ ist ein sehr mächtiges Paket zur Erstellung ansprechender Vektorgrafiken.

Mathematische Formeln

Die Formel $a^2 + b^2 = c^2$ ist kompakt und passt in den Fließtext.

Die Formel $x := \frac{1}{1+\frac{1}{1+\frac{1}{2}}}$ sprengt hingegen die Zeilenhöhe, was innerhalb eines größeren Absatzes sehr unschön aussehen kann, wie man hier sieht.

Besser: Einzeilige Formeln absetzen und zentrieren:

$$x := \frac{1}{1+\frac{1}{1+\frac{1}{2}}}$$

Und mehrzeilige Formeln? Bitte nicht so:

$f(x) = x^2 + 2x - 7$
$f'(x) = 2x + 2$
$f''(x) = 2$

Viel besser: Abgesetzt und ausgerichtet:

$$\begin{aligned} f(x) &= x^2 + 2x - 7 \\ f'(x) &= 2x + 2 \\ f''(x) &= 2 \end{aligned}$$

Formeln können auch nummeriert werden:

$$\sum_{i=0}^{n} x^i = \frac{1-x^{n+1}}{1-x} \tag{1}$$

$$\sum_{i=0}^{n} \frac{1}{2^i} = 2 - \frac{1}{2^n} \tag{2}$$

…um sie später zu referenzieren: Formel (2) ist ein Spezialfall von Formel (1).

Eine Funktion:

$$\begin{aligned} f : \mathbb{R} &\longrightarrow \mathbb{R} \\ x &\longmapsto x^2 \end{aligned}$$

Geht auch mit Fallunterscheidung:

$$\begin{aligned} \operatorname{sgn} : \mathbb{R} &\longrightarrow \mathbb{R} \\ x &\longmapsto \begin{cases} 1, & \text{falls } x > 0 \\ 0, & \text{falls } x = 0 \\ -1, & \text{falls } x < 0 \end{cases} \end{aligned}$$

Eine Matrix:

$$A := \begin{pmatrix} 1 & a & a^2 & a^3 \\ 1 & b & b^2 & b^3 \\ 1 & c & c^2 & c^3 \\ 1 & d & d^2 & d^3 \end{pmatrix}$$

Wichtig! Niemals Fließtext ohne Befehle wie `\text{...}` in Formel-Umgebungen setzen:

$$x_{Staffel}$$

Das führt zu Problemen beim Zeichenabstand und sieht komisch aus.

Beispiel für direkt eingegebenen Java-Code:

```java
import java.util.*;
public class Hallo{
  public static void main( String[] args ) {
    System.out.print("Hallo Welt!");
  }
}
```

Beispiel für direkt eingegebenen C-Code:

```c
#include <stdio.h>
int main(int argc, char** argv){
  int i;
  for(i = 0; i < argc; i++){
    printf("%s \n", argv[i]);
  }
  return 0;
}
```

Beispiel für Java-Code, der aus einer eigenen Datei eingebunden wird:

```java
package pst.gcsim.GarbageCollectors;
import java.util.*;

/**
 * Klasse zur Realisierung eines Mark-Sweep-Kollektors.
 */
public class MarkSweep<T extends Allocator> extends GarbageCollector<T> {
  public static final int MS_WHITE = 0;
  LinkedList<HeapObject> toScan;

  public MarkSweep(CollectionController<T> controller) {
    this.toScan = new LinkedList<>();
    this.controller = controller;
  }

  /**
   * Ausführung der Bereinigungsphase (ohne Animation).
   */
  void sweep() {
    controller.getObjects().sort(new AddressComparator());
    // Verhinderung von ConcurrentModificationExceptions
    ArrayList<HeapObject> toRemove = new ArrayList<>();
    for (HeapObject obj : controller.getObjects())
      if (obj.getMark() == MS_WHITE) toRemove.add(obj);
      else obj.setMark(MS_WHITE);
    toRemove.forEach(obj -> {
      controller.getAllocator().free(obj);
      controller.getObjects().remove(obj);
    });
  }
}
```

Beispiel für C-Code, der aus einer eigenen Datei eingebunden wird:

```c
#include <stdio.h>
int main(int argc, char** argv){
  int i;
  for(i = 0; i < argc; i++){
    printf("%s \n", argv[i]);
  }
  return 0;
}
```

Beispiel für Inline-Java-Code: `this.toScan = new LinkedList<>();`

Beispiel für Inline-C-Code: `int main(int argc, char** argv)`