UNIVERSITÄT ZU LÜBECK
INSTITUT FÜR
THEORETISCHE INFORMATIK

Template for the LATEX Class "uzl-thesis" for Bachelor's and Master's Theses Written at the University of Lübeck

Vorlage für die LATEX-Klasse »uzl-thesis« zur Nutzung bei Bachelor- und Masterarbeiten an der Universität zu Lübeck

Bachelorarbeit

verfasst am
Institut für Theoretische Informatik

im Rahmen des Studiengangs
Informatik
der Universität zu Lübeck

vorgelegt von
Max Mustermann (alias Till Tantau)

ausgegeben und betreut von
Prof. Dr. Petra Wichtig-Wichtig

mit Unterstützung von
Harry Hilfreich

Lübeck, den 1. Januar 2021

Eidesstattliche Erklärung

Ich erkläre hiermit an Eides statt, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Max Mustermann (alias Till Tantau)

Zusammenfassung

Es ist nicht leicht, eine Abschlussarbeit so zu schreiben, dass sie nicht nur inhaltlich gut ist, sondern es auch eine Freude ist, sie zu lesen. Diese Freude ist aber wichtig: Wenn die Person, die die Arbeit benoten soll, wenig Gefallen am Lesen der Arbeit findet, so wird sie auch wenig Gefallen an einer guten Note finden. Glücklicherweise gibt es einige Kniffe, gut lesbare Arbeiten zu schreiben. Am wichtigsten ist zweifelsohne, dass die Arbeit in gutem Deutsch oder Englisch verfasst wurde mit klarem Satzbau und gutem Sprachrhythmus, dass keine Rechtschreib- oder Grammatikfehlern im Text auftauchen und dass die Argumente der Autorin oder des Autors klar, logisch, verständlich und gut veranschaulicht dargestellt werden. Daneben sind aber auch gut lesbare Schriftbilder und ein angenehmes Layout hilfreich. Die Nutzung dieser LaTeX-Vorlage hilft der Schreiberin oder dem Schreiber dabei zumindest bei Letzterem: Sie umfasst gute, sofort nutzbare Designs und sie kümmert sich um viele typographische Details.

Abstract

It is not easy to write a thesis that does not only advance science, but that is also a pleasure to read. While the scientific contribution of a thesis is undoubtedly of greater importance, the impact of *writing well* should not be underestimated: If the person who grades a thesis finds no pleasure in the reading, that person are also unlikely to find pleasure in giving outstanding grades. A well-written text uses good German or English phrasing with a clear and correct sentence structure and language rhythm, there are no spelling mistakes and the author's arguments are presented in a clear, logical and understandable manner using well-chosen examples and explanations. In addition, a nice-to-read font and a pleasing layout are also helpful. The LaTeX class presented in this document helps with the latter: It contains a number of ready-to-use designs and takes care of many small typographical chores.

Acknowledgements

This is the place where you can thank people and institutions, do not try to do this on the title page. The only exception is in case you wrote your thesis while working or staying at a company or abroad. Then you should use the Weitere␣Unterstützung key to provide a text (in German) that acknowledges the company or foreign institute. For instance, you could use texts like »Die Arbeit ist im Rahmen einer Tätigkeit bei der Firma Muster GmbH entstanden« or »Die Arbeit ist im Rahmen eines Forschungsaufenthalts beim Institut für Dieses und Jenes an der Universität Entenhausen entstanden«. Do not name and thank individual persons from the company or foreign institute on the title page, do that here.

Contents

1

Introduction

Writing a bachelor's or master's thesis is not easy.[1] You must *research* the thesis's topic scientifically – and you must do this well. You must *describe* your research and the results – and you must use not just any words, but those that are used in the scientific community. Finally, you must *write* everything *down* – by creating an electronic document that is a pleasure to read.

It is the last item where this text may help: It is, first, a *template* that you, dear student, can copy and then modify when writing your thesis. Of course, you still have to write the text, but the template will take care of numerous technical details for you. As a teaser, have a look at Listing 1.1 on the following page, which shows the code for the "hello world of theses"[2] and which already produces a PDF file with five pages of so-called front matter (like the title page, the abstract or the table of contents) and already four pages of actual content – not bad for a single page of code.

This template document is a LATEX document that uses the UZL-THESIS document class. This means that in order to work with it, you need to use Donald Knuth's TEX text processing system (Knuth, 1986), Leslie Lamport's LATEX extension of TEX (Lamport, 1994) and my (that is, Till Tantau's) UZL-THESIS document class. In particular, you will need to learn LATEX if you have not already done so (definitely a good idea anyway).

Some students may wonder at this point whether this text applies to them at all since they do not intend to (or perhaps even may not) use LATEX for their thesis. However, while these readers can safely skip the technical details of how the UZL-THESIS class is used, I would like to urge them (and, of course, everyone else) to read Chapter 3, starting on page 36: In this chapter, I explain my views on "how to write a good thesis" and try to give as many practical hints as possible that anyone attempting to write a thesis will hopefully find useful – independently of which text processing tool they use.

The "hints" given in Chapter 3 address many of the problems that I see students struggle with when they write their thesis. Of course, I cannot give a magic recipe for creating a scientific breakthrough. But I *can* give you hints on how to put a breakthrough into

---

[1] Neither is writing a PhD thesis, but this document does *not* concern them. It is intended *only* as a template for bachelor's and master's theses written at the University of Lübeck. When you write a PhD thesis, you are invited to find your own style.

[2] In computer science, a "Hello World" program is a minimal program in a given programming language that just prints these two words.

Listing 1.1: Minimal LaTeX manuscript that generates a bachelor's thesis using the UZL-THESIS class. The manuscript has to be processed twice using lualatex, followed by a run of bibtex, followed by a run of lualatex once more.

```
\documentclass[english, version-2022-01]{uzl-thesis}

\UzLStyle{computer modern oldschool design}

\UzLThesisSetup{
  Bachelorarbeit,
  Verfasst = {am}{Institut für tolle Forschung},
  Titel auf Deutsch = {Hallo Welt},
  Titel auf Englisch = {Hello World},
  Autor = {Max Mustermann},
  Betreuerin = {Prof. Dr. Petra Wichtig-Wichtig},
  Studiengang = {Irgendwas mit Tieren},
  Datum = {1. Juli 2022},
  Abstract = {It is about saying ``hello'' to the world.},
  Zusammenfassung = {Es geht darum, der Welt »Hallo« zu sagen.},
  Numerische Bibliographie
}

\begin{document}
  \chapter{Introduction}
  \section{Contributions of this Thesis}
  This thesis says ``Hello World!'', see also \cite{Kernighan1974}.
  \section{Related Work}
  There are many hello world programs.
  \section{Structure of this Thesis}
  In Chapter~\vref{chapter-main}, we say hello.

  \chapter{Main Chapter}
  \label{chapter-main}
  Hello World!

  \chapter{Conclusion}
  Saying hello world is quite easy.

  \begin{bibtex-entries}
    @TechReport{Kernighan1974,
      author = {Brian Kernighan},
      title = {Programming in C - A Tutorial},
      institution = {Bell Laboratories},
      year = {1974}
    }
  \end{bibtex-entries}
\end{document}
```

words that other people understand and will like to read – and, hopefully, will like to reward with good grades.

Please be aware that the views expressed in Chapter 3 are *my* views and some of them may not be shared by other professors and, more importantly for you, they may not be shared by your adviser – who happens to be the person who will grade your thesis. This means that you better *always listen to your adviser* and do what she or he asks you to do.[3] The excuse "but Professor Tantau writes that..." may be flattering to me, but it will not get you high grades.

So, *always listen to your adviser.* You will read this again later on. Repeatedly.

1.1 Contributions of this Thesis

This document is both intended as a template that students can easily and conveniently adapt to their own needs and as a trove of tips and tricks on how to write a good thesis. The template as well as the UZL-THESIS class have been written by Till Tantau and they reflect – to a certain degree – my personal choices and many of the recommendations may differ from those by other people. However, one recommendation of mine trumps those made by anyone else: *Always listen to your adviser!*

1.2 Related Work

Concerning the first aspect of this text, namely the fact that it serves as a template for bachelor's and master's theses written at the University of Lübeck, let me remark that there are several older templates and styles. However, this class is the official (new) one and should be used rather than any other one.

Concerning the set of recommendations I make on how to write a thesis, first note that there are *many* books on, well, "how to write a thesis". You are *very* cordially invited to go to the library and actually read one of them (reading books *is* part of being a scientist, by the way). My personal favorite is *The Craft of Scientific Writing* (Alley, 1996). Second, note that the hints and observations I make are *my* observations and may not be shared by everyone. Of course, I can make a very good case on why you really should follow the recommendations *I* make – but remember that you should *always listen to your adviser!*

1.3 Structure of this Thesis

This thesis[4] consists of two main chapters: Chapter 2 describes how the UZL-THESIS LaTeX class is used on a technical level. This chapter starts with the technical details of how you

---

[3] If your adviser thinks the thesis should be typeset using a typewriter font with double line spacing and all headlines should be in pink, then I may (very) strongly disagree with that, but you do not have that luxury and you just typeset everything in double line spacing pink typewriter.

[4] Actually, "this text" would be more appropriate since this is obviously not a real thesis. But this is what you would write in a real thesis at this point.

setup the TEX work-flow in conjunction with the class (where to install it and which programs to use), but the bulk of the chapter is taken up by the different aspects of using that class – like how bibliographies are created or how math text should be written. The explanations only try to highlight what is important and different when using the UZL-THESIS class; they are not intended as a complete introduction to LaTeX. In Chapter 3, I then list the many small and big things you should consider and take care of when writing a thesis. I will explain how long the different parts should be, I will sketch why the abstract, the introduction and the conclusion all summarize the main part of the thesis, but still all three need to be written, I will explain why you should write "we will show that" and not "I will show that" but "I believe that" and not "we believe that" and I will give recommendations on many other topics. But of course, whatever you read in the following, remember that you must *always listen to your adviser!*

2

Guide to the Thesis Class

In this document, the *thesis class* refers to the LaTeX document class UZL-THESIS (the file uzl—thesis.cls to be precise), which was written by Till Tantau (that would be me, despite the "Max Mustermann" from the title page) to help students of the University of Lübeck when they write their bachelor's or master's thesis. In LaTeX-speak, a *document class* dictates the basic appearance of a document and this is exactly what the thesis class does: it sets up the appearance of the document in such a way that it conforms (perfectly) with the regulations of the University of Lübeck.

In addition to just formatting a thesis correctly, the thesis class also provides a number of extra commands and includes a number of packages that I have decided should be the standard at the university. This means that most students will need to load very few (typically no) extra packages for their thesis.

In this chapter, I explain all the "public" parts of the thesis class, that is, all those aspects and configuration options that students can and should use. There are also some internal keys and commands that an adventurous student might try to fool around with. Don't.

2.1 Installing and Setting Up the System

In order to write your thesis using the thesis class, you first need to setup things on your system appropriately. With a bit of luck, things have already been setup correctly for you, otherwise you will need to follow the following steps:

First, since the thesis class is based on LaTeX, you need a working LaTeX installation on the system on which you write your thesis. This "system" can be a laptop or one of the university's computers or even a cloud-based service; whatever it is, TeX needs to be installed on it. This is done using so-called *distributions* and they depend on the operating system and the details of your system. With a bit of luck, you will already have a full distribution installed, otherwise you need to get it from the net and install it. This is usually a rather simple process, but it may take a while (a full installation needs several gigabytes). I make no recommendation on which TeX distribution to use, but usually "simpler is better" and any contribution should be based on a recent version (at least 2015) of TeXlive. I have tested the class extensively with the TeX installation at the university

computer labs, with TEXlive 2015 and later, with MikTEX and with Debian-based Linux distributions (where you may need to install the additional TEX font package, though).

Second, you will need an editor for writing the text of the thesis (sometimes this text is called the *manuscript;* it is the file that TEX processes and turns into the final PDF file). This editor needs to be a simple editor that is well-suited for handling TEX files. Once more, I make no recommendations on which editor you should use: I use EMACS, but that is somewhat oldschool. Use any editor you feel comfortable with.

Third, you will actually have to run TEX on your manuscript. How this is done, exactly, will be discussed in the next section, but at this point it is important that by "running TEX" I actually mean "running the lualatex program": There are several LATEX programs out there, such as the original latex, the more recent pdflatex and the even more recent lualatex. For the thesis class, you *must* use lualatex. While this program will almost certainly be already installed as part of the TEX distribution you use, you may have to change the settings of the editor you use so that it runs lualatex rather than pdflatex.

It is well worth investing the time for this change, not only because of the UZL-THESIS class, but you should switch to lualatex in general if you have not yet done so.

Fourth and finally, you need a copy of the file uzl—thesis.cls together with some logo PDF files. Since you are currently reading this document, which is part of the directory containing the class file, there is a very good chance that you already have all the files you need. *But* you may still need to place them somewhere where TEX will find them. A good place for this is usually the directory that also contains your thesis manuscript (remember, that is the file that contains your thesis and ends ".tex").[5] I have carefully tried to make sure that the number of files needed for the thesis class is as small as possible and that it should work out-of-the-box on most systems.

Test whether everything works by running LuaLATEX on this file, that is, by running:

```
lualatex Template_for_the_LaTeX_Class_uzl-thesis.tex
```

If there is a problem or error, do not be too shy to ask somebody to help you with the setup!

2.2   Running LATEX and BibTEX

With the system up and running, you can start writing the manuscript. The work-flow is the following:

1. Make a copy of Template_for_the_LaTeX_Class_uzl—thesis.tex and rename the copy to something like, for instance, mythesis.tex.[6] This will be the *manuscript* of your thesis.

---

[5] If you are more TEX savvy, you may place the directory with the uzl—thesis.cls in a local TEXMF tree. However, in the long run, it is advisable to keep all files needed for typesetting you thesis together – namely in the directory of the manuscript.

[6] Note that the file is encoded in UTF8 encoding (unless you study computer science, you really do not wish to know the details) and that the encoding should stay that way. Fortunately, most editors will use this encoding automatically, so this should not be a problem.

2. Replace all parts of the text (like this one) with your own text. Use your favorite editor for this.
3. Now run lualatex on the manuscript.
4. If there are *any errors*, then *correct them first.* It is usually a very bad idea to ignore the errors. Ask someone if you do not know how to fix an error!
5. After you have successfully run LaTeX at least twice, run bibtex. Once more, fix any errors before proceeding.[7]
6. Then run lualatex once more.
7. You should now have a correct PDF file.

Congratulations on creating a first version of your thesis. As you may already have noticed when looking at Listing 1.1 on page 2, the thesis manuscript contains a rather prominent block in the preamble where the \UzLThesisSetup command is used. In the following, we have a closer look at this command; but, first, we have a look at the very first line of the listing, namely the class options.

2.3   Specifying the Language and Version

Your manuscript must start with

```
\documentclass[german, version-2022-01]{uzl-thesis}
```

or with

```
\documentclass[english, version-2022-01]{uzl-thesis}
```

depending, of course, on whether your thesis is written in German or in English. Note that no other languages are supported (or allowed) and that you *must* specify *exactly* one of the two language options[8].

You *must* also provide the class version key[9] and note that this is the version and date of the thesis *class* and not the date when you turn in *your* thesis. The key ensures that when the thesis class is changed in the coming years, using a new version together with your (original) manuscript will still produce the same PDF as it did when you wrote your thesis. Just leave this class version key as it is.

2.4   Specifying the Overall Design

The way a thesis looks – its *style* or *design* – is a matter of taste; but only to a certain degree. While there are many things that are "not open to discussion" like the fact that the main

---

[7] There is also the option of using biber, see Section 2.13 on page 33 for details. If you have no idea what biber is, just ignore this remark.
[8] Do not worry about that fact than an English thesis will contain a German abstract text, which needs special hyphentation – this is taken care of automatically. As I promised, the thesis class takes care of many technical details that you may not have thought about.
[9] In computer science parlor, a "key" is a text like version—2022—01 that is specified in lists and that causes some configuration to be chosen.

font should have a certain minimum (and maximum) size and should be easy to read, there is no reason why any *particular* font *must* be used in a thesis or why a, say, centered layout is better than a flushed-left layout.

With the thesis class, you can quite easily change the way the thesis looks – but you are limited to those choices that I have vetted and that I know work well together. While this is definitely a better approach than the one I see elsewhere (where each student copies layout code from someone else, neither understanding what it does nor what it is actually for, and then hopes for the best), I restrict your freedom when it comes to styling the thesis: Even if you really think that pink headlines and a slab serif as main font are way cooler that the designs I chose, you will just have to live with the options I provide. I guess that 99% of all students will be more than happy to leave the design process to me, but if you are one of the 1%, you may complain fiercely at this point and may then just pick one of the styles I chose for you like everyone else.

Choosing a Design

To pick a design, you call one command in the preamble (that is the part following the \documentclass before the actual thesis starts with the {document} environment):

\UzLStyle{⟨*style*⟩}

where ⟨*style*⟩ is one of the following (all in lowercase, as in \UzLStyle{alegrya␣basic␣ design} for instnace):

computer modern oldschool design The look of this design mimics the "classical" way a paper or report created with LaTeX looks like: The Computer Modern font is used, bold face fonts are used for headlines, only black and white are used instead of colors. This design reminds me of older scientific documents, especially from the computer science community where LaTeX was used very early.

computer modern basic design A slightly less "oldschool" version of the previous design. It is still a classic design in the sense that it uses the Computer Modern font and that it still has this "good old LaTeX" look, but some more modern aspects (like colors!) have been added.

Note that this design uses the Myriad font for the title page (one of the "modern aspects"), which means that this font must be installed. This font is the university's official font, which is why it is appropriate for the title page. However, the font does not otherwise go well with the Computer Modern font, so it is used only on the title page.

computer modern scholary design In my opinion, this is the ultimate "scholary design": The thesis will look like it had been typeset by hand some 150 years ago and then printed by a university press. There is really nothing "modern" about it despite this word being part of the design's name: the word *modern* is just part of the name of the Computer *Modern* font.

pagella basic design A, well, basic design that uses the Pagella font rather than the Computer Modern font. Especially the bold face version of this font looks nicer than the Computer Modern counterpart. Also, Pagella, while still having a "bookish" look, feels a bit fresher than Computer Modern.

pagella centered design A variant of the basic Pagella design that centers all headlines. A nice alternative to the basic version.

pagella contrast design This design tries to create some visual friction by contrasting the sans serif headline font (in bold!) with the main text. I find it a visually very interesting combination.

alegrya basic design The third variant of the basic design, this time using the Alegrya font.

alegrya scholary design The Alegrya version of the previous "scholary" design. Unlike the Computer Modern version, this design does not look old, but more fresh – while still creating the impression that the text must be about a very scientific subject.

alegrya stylish design The design is quite similar to the scholary version for the Alegrya font, but with even more modern additions. "Stylish" is the word that comes to my mind.

alegrya modern design A design that uses the sans serif version of the Alegrya font for the headlines. This is a nice modern overall design.

Notes on Fonts

The different designs from which you can choose use three different main fonts: Computer Modern, Pagella and Alegrya. Of course, there are many more fonts around and even installed on your computer – but you can still use only one of these three. The reason is that there is much more to choosing and setting up a font than meets the eye: One must ensure that the font works not only for simple text, but also for complex mathematical text, that the font sizes fit and are comparable to those of other fonts, that the typewriter (the monospace) version goes well with the main font, that the small caps version is correctly selected and much more.

All of these complex setups have been carefully done for these three fonts (the setup for Alegrya alone needs 50 lines of code of the thesis class). Since it is almost impossible for a beginner (or even an advanced user) to get all of these setups right when choosing another font, you are kindly requested to refrain from doing that. Use one of the three fonts, which get installed as part of the different designs.

Further Styling

The styling mechanism actually allows you finer control over the styling of different elements of a thesis, like the font and color used for section titles, the layout of the table of contents and many more aspects. However, while you find some documentation in the comments inside the thesis class file, I do not provide a detailed documentation here since, usually, you should not configure these styles individually.

2.5   Specifying the Metadata Like Title and Abstract

You *must* use the command \UzLThesisSetup (at least) once in the preamble. This command is used to configure the so-called *metadata* prior to the first chapter of the thesis. The metadata is information "about" your thesis (hence the "meta-") and must be provided by this special command. Note that there are normally other commands in LaTeX

for providing such metadata (like \author), but you *may not use them* since all metadata is specified using the \UzLThesisSetup command. The syntax is as follows:

```
\UzLStyle{⟨key⟩ = {⟨value⟩},
         ⟨key⟩ = {⟨value⟩},
         ...,
         ⟨key⟩ = {⟨value⟩}}
```

For some keys, the =␣{⟨*value*⟩} part may be missing. You can use the command several times, but all uses must be within the preamble and usually you will put everything in a single command. There are numerous keys that can be configured with this command and most of them are required, that is, they *must* be set (exactly once). The order in which you set them is, however, not important.

The keys that you *must* set are the following (note that most of them are in German; this is to remind you that you *must* provide German texts as values for these keys *even in an English thesis* since the university's guidelines insist that the text on the title page must be in German):

Bachelorarbeit / Masterarbeit   You must specify exactly one of these two keys. Note that this thesis class is *not* meant for PhD theses.

Verfasst   Specify the institute or clinic where you wrote the thesis. Note that this must be the official name of one of the institutes and clinics of the university (sometimes even the supervisors do not seem to know the real name of their own institute...). The key takes two arguments as value (it is a rather special key): The first is the word am (for institutes) or the words an␣der (for clinics), the second is the name of the institute or clinic. Here are two examples:

```
Verfasst = {am}{Institut für Theoretische Informatik}
```

or

```
Verfasst = {an der}{Klinik für Hals-, Nasen- und Ohrenheilkunde}
```

It would be wrong to use "Klinik für HNO" instead. If in doubt, consult the "Satzung über die Institute und Kliniken der Universität zu Lübeck" from the university homepage. It would be *utterly wrong* to write "Universitätsklinikum Schleswig-Holstein" here, by the way, since that institution is not part of the university.

Note that if you have written your thesis externally (like abroad or at a company), you *still* have a supervisor at the University of Lübeck and she or he is *still* affiliated with an institute of the university. It is *that* institute or clinic that must be named here.

Logo-Dateiname   If the institute has provided you with its own logo as a PDF file, you can specify the file name here. Note that only those logos are allowed here that look like the university logo (see Figure 2.6 on page 28), but with the name of the institute added. Also note that the logo PDF file should have the exact same size and scaling as the normal logo.

Titel auf Deutsch   Provide the German title here. *Please* try to avoid spelling mistakes at least here. Wrong capitalization *is* a spelling mistake in German, by the way: Do *not*

entitle your thesis "Untersuchungen zur Schnellen Simuluation von Turing Maschinen" – that is two spelling mistakes! You must use "Untersuchungen zur schnellen Simulation von Turing-Maschinen".

Titel auf English Provide the English title here. Note that there are *also* rules for capitalization in English, but they are a bit more flexible. For more information on when to use uppercase and when to use lowercase in English, see Question 3.25 on page 47.

Studiengang Make sure the name of degree programme is correct and not some abbreviation or some incorrect variant. For instance: "Medizinische Ingenieurwissenschaft", but not "MIW"; "Medizinische Informatik", but not "Medizin-Informatik"; "Informatik", but not "Informatik (SSE)".

Use German names for German programmes and English names for English ones, so "Infection Biology", not "Infektionsbiologie" and also neither "Infection-Biology" nor "Infection biology". For programmes that have a German bachelor programme and an English master programme, use the German name for a bachelor's thesis and the English name for the master's thesis.

Autor / Autorin Both keys have exactly the same effect. In either case, that would be you, I sincerely hope. Make sure you spell your own name correctly. It is slightly embarrassing to get it wrong (I saw that on a thesis, once; so, you have been warned). If you have more than one first name, you can include them.

For your own name, do not add things like "BSc" or "BSc cand." (especially the latter is slightly ridiculous). For the adviser, you should normally include "Prof. Dr." or "PD Dr." (ask your adviser, what is appropriate), but nothing more (so no "Univ.-Prof. Dr. Dr. h. c. mult." unless your adviser insists).

Betreuer / Betreuerin Both keys have exactly the same effect. Note that this key specifies the *adviser*, that is, the professor who officially supervised the thesis. If there is also an assistent of the professor who helped (typically a lot), use the key Mit␣Unterstützung␣von (see below) to thank that person. If the thesis was mainly written externally at some company or another institute, point this out using Weitere␣Unterstützung – but note that the adviser will *still* officially be a professor of the University of Lübeck and you must name her or him, here.

Datum The date you turn in your thesis. Do *not* be late. Format the data in the "traditional German way", that is, the first of June 2020 must be formatted as "1. Juni 2020". Do *not* write any of "01. Juni 2020" or "01. 06. 2020" or "1.6.2020" or "1. 6. 2020" or "2020-06-01".

Abstract Provide the (whole) English abstract of your thesis here. It may have more than one paragraph, but it should not. You will find more information on how to write an abstract in the answers to Questions 3.6 and 3.7 on page 39.

Zusammenfassung Provide the German abstract here. It should be a faithful, but not word-to-word translation of the English one (or *vice versa*).

Alphabetische Bibliographie /

Numerische Bibliographie You must use exactly one of these two keys. They select which bibliography style is used throughout the thesis. If you write a thesis in a degree programme of the natural sciences (like Molecular Life Science), you *must* use the first key. Otherwise, you can use either one. You can find more information on the bibliographies in Section 2.13 on page 33.

In addition to the above keys, all of which are required, there are also some optional keys:

Mit Unterstützung von If there has been an assistant that helped the supervisor (a lot), name and thank this person here. Do not list everyone at the institute.

Weitere Unterstützung You may (but need not) provide a text (in German) that acknowledges a company or an institute that is not part of the University of Lübeck where the thesis was mainly written. For instance, you could use texts like "Die Arbeit ist im Rahmen einer Tätigkeit bei der Firma Muster GmbH entstanden" or "Die Arbeit ist im Rahmen eines Forschungsaufenthalts beim Institut für Dieses und Jenes an der Universität Entenhausen entstanden". Do not name and thank individual persons from the company or foreign institute on the title page, instead use the Acknowledgements key for that.

Acknowledgements You can provide a longer paragraph in which you thank all the nice people who helped you with the thesis. This text will not be shown on the title page, but will be shown rather prominently before the table of contents. It is usually a good idea and a nice touch to name everyone who had even a remote positive influence on your thesis.

2.6   Including Default and Optional Packages

Following the \UzLThesisSetup command, you can start loading additional packages that you may need for typesetting your thesis. For instance, you may wish to load special libraries for drawing finite automata or you may wish to load a special package for doing fancy things with tables. Use LaTeX's \usepackage and TikZ's \usetikzlibrary commands as usual for this.

However, the uzl-thesis class loads many packages already and sets them up in certain ways. This has two effects:

1. You often do not need to load additional packages at all for your thesis since for almost all common problems a suitable package will already have been loaded, configured and patched by me.

2. You cannot load a number of packages, because they clash with the packages loaded by uzl-thesis. For instance, you cannot use the pstricks package since TikZ is already loaded, you cannot use natbib since biblatex is loaded and so on. For all cases where this happens, I had a closer look at the alternatives and then explicitly chose the package that is now loaded. Even if you like another package better, you will have to live with the choice I made.[10]

The following packages are preloaded:

amsmath, amssymb and amsthm These package are all part of the $\mathcal{A}_{\mathcal{M}}\mathcal{S}$Math family. In particular, the whole management of theorem environments is built on top of these

---

[10] If you really want to change these choices, finish your studies quickly, finish your PhD quickly, become a postdoc, do amazing research, become a professor at the University of Lübeck and get voted as Dean of Studies. Then you can freely change which packages are preloaded.

packages, see Section 2.9 for more details. You can use all commands and symbols provided by these packages, see their documentation for more information (The LaTeX Team, 2017). Any package that is incompatible with $\mathcal{A}_{\mathcal{M}}\mathcal{S}$Math cannot be used.

babel This package is loaded and the main and secondary languages are setup based on the class options (german or english). You can use the commands provided by BA-BEL, see its documentation for details (Braams and Bezos López, 2019).

biblatex This package is used for the whole bibliography management. There are several alternatives, but this one of the places where I made a choice that it not open to discussion. Fortunately, BIBLATEX is very powerful and you can use its powerful commands in your thesis. See its documentation (Kime and Lehman, 2018), but also Section 2.13.

fontspec This package is used for the whole font setup in conjunction with LuaLaTeX (and, once more, this is not open to discussion). You can use the powerful commands of this package *but* you will only rarely need to do so. Mostly, the font styling is taken care of via global styling parameters, see Section 2.4 on page 7 for more details.

geometry This package is internally used to setup the page layout. *Under no circumstances are you allowed to mess around with the margins.* They are perfect the way they are.

There is a case to be made that there should be an option for slightly shifting the pages horizontally depending on whether an odd or an even page is printed: When pages printed on both sides (double-sided) are bound as a book, it is desirable that the inner margin – where the binding is done – is larger. However, in my experience, theses are very often read on electronic devices these days and, then, it is distracting when the text "jumps around" from page to page. For this reason, the default is a fixed layout; but you *may* consider doing fancy things for a print version that is to be bound as a book. Normally, however, this will not be necessary since the margins are large enough that binding is still possible without any shifting.

hyperref This is a standard package for creating all the different links inside the thesis. You can also use its facilities for creating links to external material, see the HYPERREF package's manual (Rahtz and Oberdiek, 2018).

Note that I took great care to absolutely ensure that there will be *no* borders in the PDF file around hyperlinks. The default settings of the HYPERREF package for these borders are among the worst and most ugly pieces of typography I have ever seen. Whoever made them the default should be ashamed of themselves. That means, of course, that *you* should not even think of fiddling around with these settings.

listings This is one of several possible packages for creating listings of source code (like the one in Listing 1.1). I have chosen this package and configured it carefully for your use and strongly recommend that you use it (instead of any other package). You will find much more information on it in Section 2.10 on page 20.

tikz This is a powerful package for creating graphics right inside the LaTeX document (I happen to be the author, but I do not include it out of vanity but because it *is* the standard package for creating graphics inside LaTeX documents these days). Some *TikZ* libraries are preloaded (arrows.meta, positioning and shapes) and a number of styles are predefined that you should use, whenever possible, see Section 2.12 on page 27 for more details.

url Another standard package that is included for typesetting URLs more nicely. This is

mainly useful for the bibliography, where it is used automatically. See its documentation for details (Arseneau, 2013).

varioref This is a standard package for creating nicer internal cross-references and you can use its facilities freely. See its manual (Mittelbach, 2016) and also Section 2.13.

In addition to the above, the packages CSQUOTES, ETOOLBOX and FILECONTENTS are also included, but only for internal use. It is unlikely that you will need their facilities.

2.7   Structuring the Thesis Using Chapters and Sections

Once you have specified all metadata and loaded all packages that you need and like, you start with the so called *body* of your thesis. Normally, this would start with commands like \maketitle and \tableofcontents and other commands to create the so called *front matter*, which is everything before your actual text. However, with the UZL-THESIS class, you do *not* use any of these commands. Rather, the thesis class will insert all necessary front matter automatically at the beginning of the thesis – and it will do so in the right order and with the right page numbering and so on. This means that you can (and must) directly start the {document} enivornment with the first chapter, which will be the introduction. Symmetrically to the front matter, the thesis class will also insert the bibliography automatically at the end. (In particular, you cannot have a thesis without a bibliography, but that would not be very scientific anyway.) Once more, please consult Listing 1.1 on page 2 for an example of how the overall structure of the thesis must look like.

Chapters and Sections

The main part of your thesis consists of *chapters*[11] (using the \chapter command) and those in turn consist of *sections*[12] (using the \section command). The commands take the usual arguments as in other document classes.

Subsections

It is also possible to use the \subsection command to further structure sections, but the text will not be shown in the table of contents. The reason is that it is generally a bad idea to nest structures too deeply. Richard Feynman is renown for his *Feynman Lectures* in which he introduces the fundamentals of physics in three thick volumes using only chapters and sections. If Richard Feynman can explain all of physics using only two levels of structural nesting, so can you in a hopefully somewhat shorter thesis. So, while is acceptable to use \subsection to structure sections, do not try to force the titles of the subsections into the table of contents and do not try to reference them.[13] Also, do not even think of using \subsubsection.

---

[11] In a German thesis, use the word "Kapitel" to refer to a chapter.
[12] In a German thesis, use the word "Abschnitt" to refer to a section.
[13] Having said this: If it is absolutely necessary that you need subsections in the table of contents, try saying \setcounter{tocdepth}{2} at the beginning of your document.

Lists of Figures and Tables

In some books you will find lists of all figures and lists of all tables as separate lists. Frankly, I have never looked at any of them and consider them perfectly superfluous (and a very unscientific survey that I conducted shows that all of my colleagues agree). I guess they are mainly included in books (and some theses) because LaTeX makes it easy to generate them and because they make theses appear longer than they actually are.

I strongly recommend that you do *not* include such lists in your thesis. However, in case your advisor *really insists*, you can say the following (or just one of them, in case you have no figures or no tables):

```
\UzLThesisSetup { Abbildungsverzeichnis, Tabellenverzeichnis }
```

This will add the named lists after the table of contents.

Appendices

It is possible to add appendices using the \appendix command followed by a \chapter command. The \appendix command will automatically insert the bibliography first. You will find an example in the comments of this template, but I will not explain the details further at this point since I would like to discourage appendices: In a normal thesis, the main text should explain everything there is to explain and should also include and relevant data, tables, listings and proofs. An appendix is usually only needed if there is a lot of experimental data and the readers is invited to have a closer look at that data – but, again, it is then the job of the author to point out the important features in the main text. I believe it is better not have an appendix in the thesis and to move all raw data and all source code into the *electronic supplementary material*, which is turned in along with the thesis text.

Note, however, that the above arguments only apply to your bachelor's or master's thesis: Many scientific *papers* actually need an appendix for various reasons.

2.8   Typesetting Basics

The majority of your thesis will of course consist of "normal text" like this one. In LaTeX you simply enter this text and the thesis class does little to interfere with this process. There are only a few things you should keep in mind and be aware of:

Unicode Characters

The version of LaTeX used by the thesis class, LuaLaTeX, uses the Unicode throughout. This means, in particular, that you can and should enter your main text using the correct and appropriate Unicode characters. This is true, in particular, for quotation marks. Supposing you find the characters on your keyboard, you should write, for instance, in a German text Er␣sagte,␣»Hallo!« where you use the correct Unicode characters directly.[14]

---

[14] If you are interested, here is a bit of historical context: When Donald Knuth first designed TeX, the Unicode had not even been invented yet. However, Knuth already wanted to typeset for instance the German

Enumerate and Itemize

You can use the {enumerate} and {itemize} environments as usual, but I have changed the vertical spacing compared to the standard LaTeX styles (nobody understands why these environments insert enormous vertical spaces between items in standard LaTeX). Do not mess with the spacing.

Note that you can nest the environments, but only up to one level. It is generally a bad idea to nest the environments and it is definitely a bad idea to nest them repeatedly. Just do not do that.

Line Breaking

The lines of a paragraph are automatically broken up by TeX and you usually do not have to worry about line breaking and hyphenation. I have also changed the way the titles of chapters and sections are broken both in the main text and in the table of contents compared to standard LaTeX so that they will usually look nice. However, there are cases where the line breaks can be improved: Consider for instance a chapter title like "Review of the System Setup and the Experiment Design" and suppose that the last word of this title ("Design") no longer fits on the line. Then LaTeX will do a line break between "Experiment" and "Design" – but it would be much better to have the line break *before* "Experiment" so that the words "Experiment Design" stay together. You can achieve this by using a tilde instead of a space between the two words: A tilde is used in TeX to indicate a "nonbreakable space", which is exactly what we want. Thus, you would write:

```
\chapter{Review of the System Setup and the Experiment~Design}
```

The same is also true for the title of the whole thesis: Use the tilde in all places of the title where *no* line break should occur. For instance, the title of the present document was specified as:

```
\UzLThesisSetup { Titel auf Deutsch = {
    Vorlage für die \LaTeX-Klasse »uzl-thesis« zur Nutzung bei
    Bachelor-~und Masterarbeiten an der  Universität~zu~Lübeck
} }
```

Here, a nonbreakable space was inserted after "Bachelor-" to ensure that no line breaking is done exactly there since because of the hyphen a reader might read "Bachelorund". Two further nonbreakable spaces ensure that "Universität zu Lübeck" is always on one line.

---

umlaut "Ä" in his texts. His solution was to design a special command (\") that puts the umlaut on top of a character, so "Fräulein" would be typeset as Fr\"aulein. Clearly not an optimal solution for longer German texts, but it was a simple solution for individual words – and you will still see it a lot in BIBTEX files.

These days, you should no longer write things like Fr\"aulein in your texts, but just write Fräulein. This is not only true for umlauts, but for all kind of symbols for which the Unicode offers individual symbols. For instance, you can and should also use the (correct) symbols for quotation marks – there are Unicode characters for all of them, see also Question 3.27 on page 48.

2.9   Typesetting Mathematics

TEX is great at typesetting mathematics; indeed, it is nowadays the gold standard for the correct typesetting of mathematical text. However, perfectly mastering TEX's rules for mathematical text is not easy and this document cannot cover them fully; indeed, it cannot even give a full introduction and you are kindly requested to read the TEXbook (Knuth, 1986). The following text only highlights the peculiarities of the thesis class with respect to mathematical text.

First, note that the thesis class preloads the $\mathcal{AMS}$math package, that is, the LATEX packages AMSMATH, AMSSYMB and AMSTHM. This means that you can and should use the commands offered by these packages, if you need them.

Predefined Mathematical Environments

The thesis class predefines a whole set of environments that you should use for mathematical statements. A typical way of using them is as in the following code:

```
\begin{conjecture}[Goldbach]
  Every even integer $n \ge 4$ is the sum of two primes.
\end{conjecture}
\begin{lemma}
  Every $n \in \{4,6,8,10\}$ is the sum of two primes.
\end{lemma}
\begin{proof}
  We have $4 = 2+2$, $6=3+3$, $8 = 5+3$ and $10 = 5+5$.
\end{proof}
```

This is then typeset as:

Conjecture 2.1 (Goldbach). *Every even integer $n \geq 4$ is the sum of two primes.*

Lemma 2.2. *Every $n \in \{4, 6, 8, 10\}$ is the sum of two primes.*

Proof. We have $4 = 2 + 2, 6 = 3 + 3, 8 = 5 + 3$ and $10 = 5 + 5$.   □

As you can see, the numbering of the environments starts anew with each chapter, but is otherwise consectutive. As someone who has to read a *lot* of mathemtical texts, I can tell you that this is the only sensible way of numbering things – which is why this numbering style is hardcoded into the thesis class. Do not mess with it.

The following environments are predefined (in a German thesis, you *still* use the following environments, they get translated automatically):

theorem A mathematical statement proved in the context of the thesis. Note that even simple statements can be theorems.

fact A mathematical statement that has already been proved in a paper. Always cite where it was proved.

lemma A mathematical statement that is only needed for the proof of another theorem. Just because a statement is not "very important" it is not automatically a lemma, but can still be a theorem. What makes a statement a lemma is the fact that it is not needed outside the context of another theorem.

corollary A theorem that follows very easily from the previous result.

conjecture A (mathematical) statement that the author suspects to be true, but does not know how to prove.

definition A (mathematical) definition.

problem The description of a problem.

example A single example. Always try to give (good) examples.

examples Use this for multiple examples in a single environment.

counterexample For a single counterexample. There is a saying that any good definition should be accompanied by at least one example and at least one counterexample.

counterexamples For multiple counterexamples.

observation A (mathematical) statement that is so simple that no proof is given and none should be needed.

remark Any remark that is so important that needs a number and can be referenced. Normal remarks can just be written as part of the text.

proof Note that this will automatically add an end of proof mark at the end.

Additionally, for some of the above there are "starred versions," which are not numbered:

example*

examples*

counterexample*

counterexamples*

observation*

remark*

Finally, there is the following environment:

claim* This is an (unnumbered) mathematical statement that is made and proven *inside* the proof of a lemma or theorem.

As two further examples of how these environments are used, here is another example:

```
\begin{problem}\label{problem-vc}
  The problem $p$-\textsc{vertex-cover} is the set of all pairs $(G,k)$,
  where $G = (V,E)$ is a graph, for which there is a set $X
  \subseteq V$ such that $|X| \le k$ and such that $\{u,v\} \in E$
  implies $\{u,v\} \cap X \neq \emptyset$.
\end{problem}

\begin{fact}[\textcite{DowneyF13}]
  The parameterized problem $p$-\textsc{vertex-cover} from
  Problem~\ref{problem-vc} is fixed-parameter tractable.
\end{fact}
```

This is rendered as:

Problem 2.3. The problem p-VERTEX-COVER is the set of all pairs (G, k), where $G = (V, E)$ is a graph, for which there is a set $X \subseteq V$ such that $|X| \leq k$ and such that $\{u, v\} \in E$ implies $\{u, v\} \cap X \neq \emptyset$.

Fact 2.4 (Downey and Fellows (2013)). The parameterized problem p-VERTEX-COVER from Problem 2.3 is fixed-parameter tractable.

"Theorem" or "Satz"?

In German, a "theorem" is traditionally called "Satz" and a "lemma" is actually a "Hilfs-satz". However, many people feel that "Theorem" sounds more scholary and, thus, use "Theorem" and "Lemma" also in German texts. In German text, the thesis class uses "Satz" and "Hilfssatz" (and "Folgerung" for "corollary") by default. If you prefer the Greek words, say

```
\UzLStyle{Greek theorem labels}
```

to change the words to "Theorem", "Lemma" and "Korollar" in German texts.

Defining New Theorem Styles

You can define additional mathematical environments using the facilities of the AMSTHM pacakge, which is preloaded and which is used for the mathematical environments of the class. Normally, you will not need this. Nevertheless, if you really do, make sure you define things the same way as in the class file (have a look there). For instance, here is how the {question} and {answer} environments are defined, which are used a lot in the second part of this document:

```
\theoremstyle{plain}
\newtheorem{question}[theorem]{Question}

\theoremstyle{remark}
\newtheorem*{answer}{Answer}
```

The reason I usually recommend against new environments is that especially begin-ners tend to define many new environments whose semantics are somewhat unclear. I have seen texts in which there are theorems, lemmas, observations, propositions, facts, corollaries, statements and more all mixed together. I sincerely doubt that even the au-thor had an idea what the exact difference between a proposition and a theorem was sup-posed to be. Try to stick to the predefined environments, they suffice.

Display-Style Mathematical Statements

Mathematical statements can either be given "inline" like $a^2 + b^2 = c^2$ here or they can get their own line as in

$$\sin^2 \alpha + \cos^2 \beta = 1 \tag{2.1}$$

Note that the equations are flushed left if you use standard commands. Indeed, the only command you should *not* use is $$, see also Question 3.21 on page 46 for more details on which environments to use.

2.10   Typesetting (Pseudo) Code

Theses from the area of computer science and related disciplines sometimes include listings of source code and of pseudo code. The thesis class preloads the LISTINGS package and configures it for you and you are kindly requested to use it for typesetting both normal code and pseudo code (that is, algorithms).

Since I reconfigured the LISTINGS package quite a bit, let me point out what you should and should not do when typesetting code:

– Use one of the environments and commands defined by the thesis class or, if your favorite language is not defined yet, define a new one using the special command \uzldeflanguageshorthand. For instance, to typeset C++ code, you write

```
\begin{C++}
void main () { std::cout << "Hi!"; }
\end{C++}
```

This has the same effect as

```
\begin{lstlisting}[language=C++, style=code]
void main () { std::cout << "Hi!"; }
\end{lstlisting}
```

In either case, the result is:

```
void main () { std::cout << "Hi!"; }
```

The style=code sets up the fonts and the correct highlighting. You should always use it, but it is selected automatically by the standard environments, see below.

– Environments like {C++} take options just like the {lstlisting} environment. Use these options as explained in the manual of the LISTINGS package to setup floats, captions, label, line numbers, the programming language and so on. However, you should *not* use the options to mess with the typesetting nor to redefined the fonts or other such evil endeavors.

– For pseudo code, use the {Pseudocode} environment. It is explained in more detail on page 22.

Language Environments

The thesis class defines environments for typesetting a number of languages. Each of these environments installs appropriate fonts and select a specific programming language for the syntax highlighting. Note that all of the environments start with a capital letter. The available environments are the following:

– The {Code} environment does no special highlighting at all. It is useful for "plain text" or code the should *not* be highlighted:

```
\begin{Code}
begin the end of the world // end the begin of the world
\end{Code}
```

yields

```
begin the end of the world // end the begin of the world
```

– The {C} environment should be used for code in the C programming language.

```
\begin{C}[
  float,
  caption = {
    The first C program from the tutorial in \cite{Kernighan1974}.
  },
  label = {lst-hello}
  ]
main( ) {
        printf("hello, world");
}
\end{C}
```

yields Listing 2.1.

Listing 2.1: The first C program from the tutorial in (Kernighan, 1974).

```
main( ) {
        printf("hello, world");
}
```

– The {C++} or {Cpp} environments should be used for C++ code. They have the same effect.
– Use {Java} for Java code.
– Use {Latex} for LaTeX code (note the spelling of the word Latex: A capital L, but otherwise lowercase letters. The command \LaTeX yields, in contrast, the word "LaTeX").
– Use {Matlab} for MATLAB code.
– Use {Pseudocode} for pseudo code and algorithms, see below for more details.
– Use {Sql} for SQL code (note the spelling once more: A capital S, but otherwise lowercase letters).

The LISTINGS package supports a large number of additional languages, but I have decided to pre-define only the above. You can easily define further languages using the following command:

```
\uzldeflanguageshorthand{⟨Environment⟩}{style=code,language=⟨language⟩}
```

Here, ⟨*Environment*⟩ should of course be the name of the environment (preferably with a leading capital letter) and ⟨*language*⟩ must be a language name that the LISTINGS package understands. As an example, here is a definition for Pascal:

```
\uzldeflanguageshorthand{Pascal}{style=code,language=pascal}
% Now you can say
\begin{Pascal}
sum := 0;
for i := 1 to 100 do
begin
  sum := sum + i
end
\end{Pascal}
```

and get

```
sum := 0;
for i := 1 to 100 do
begin
  sum := sum + i
end
```

Pseudocode Environment

The {Pseudocode} environment is special compared to the other languages: A different font is used and the spacing is different. You use it as follows:

– You write down your pseudo code and algorithms in a "literal" way, which means that you write them down inside the environment more or less the way they should look like later. Here is a simple example:

```
\begin{Pseudocode}
print ``Please enter a number below 10.''
input $n$
if $n > 9$ then
  print ``Too high!''
else
  print ``Thank you!''
\end{Pseudocode}
```

The output is then:

print "Please⨯enter⨯a⨯number⨯below⨯10."
input n
if n > 9 *then*
   print "Too⨯high!"
else
    print "Thank⨯you!"

– As can be seen in the example, you can and should put variables and mathematical expressions inside the LaTeX math environment using $'s. Indeed, the {Pseudocode} environment allows you to "escape" to LaTeX math mode at any point. I recommend writing "$n␣>␣9$" rather than "n␣>␣9" or just "n␣>␣9" or "n␣>␣9" since only the first version will get all the spacing and fonts correct. See also Question 3.22.

– Use `` and '' to start and end strings.

– Use // to start comments. There are no multi-line comments in pseudo code. The comments will not flush right. If you really, really want this, you can do tricky things with escapes and \hfill. I will not tell you how to do this.

– Some common keywords are setup and highlighted automatically. See Listing 2.2 on the next page for an example of how to add or remove words from the keyword list.

– The character sequence "<—" is redefined so that it yields a left-pointing arrow ("←"). This is usual way to typeset assignments in pseudocode: Say

```
\begin{Pseudocode}
algorithm compute_sum ($n$)
    // This function computes the sum of the first $n$ numbers.
    if $n<0$ then
        throw ``negative number''
    $s$ <- $0$
    foreach $i \in \{1,\dots,n\}$ do
        $s$ <- $s + i$
    return $s$
\end{Pseudocode}
```

to get

algorithm compute_sum (*n*)
    // This function computes the sum of the first *n* numbers.
    if $n < 0$ *then*
        throw "negative×number"
    $s \leftarrow 0$
    foreach $i \in \{1, \dots, n\}$ *do*
        $s \leftarrow s + i$
    return s

Floating Code

In TeX, "floating" text is a figure, table or listing that is not necessarily inserted at the point where it is in LaTeX source code, but at a possibly later point so that it fits nicely on the page. For listings, you can use the float key to turn the listing into a floating listing. For instance, the code

```
\begin{C++}[caption={A ...}, label={lst-1}, float, numbers=left]
#include <iostream>

void main()  // Where did the arguments go?
```

Listing 2.2: A merge implementation in pseudocode. For this example, two additional keywords were added using the option morekeywords␣=␣{append,␣remove} while one keyword got deleted using deletekeywords␣=␣to. This causes the words "append" and "remove" to be highlighted, while "to" is not highlighted.

```
procedure merge(A, B)
    C ← empty list
    while A not empty and B not empty do
    if A.first < B.first then
        append A.first to C
        remove first from A
      else
        append B.first to C
        remove first from B

    append remaining elements from A and B to C
    return C
```

```
{
    std::cout << "Hello World!" << std::endl;
}
\end{C++}
```

yields Listing 2.3 which is right behind this paragraph in the LaTeX manuscript, but which will not necessarily be right behind this paragraph in the PDF file.

Listing 2.3: A "floating" hello world program in C++. It was typeset using the C++ environment with the key caption set to the text you are reading now, the key float set to make the listing "float" and the key numbers␣=␣left to add some line numbers.

```
1 #include <iostream>
2
3 void main()  // Where did the arguments go?
4 {
5     std::cout << "Hello World!" << std::endl;
6 }
```

Note that you should always use the caption key to provide an explaining caption for floating listings. You can also use the key for non-floating listings, but I advise against that: better explain things in the main text directly before or after the listing.

Inline Listings

It is possible to "inline" listings, which just means that code is displayed in the middle of a normal paragraph and not in a separate block. You use the commands of the same name as the corresponding language environments for this: For instance, while the environment {Java} is used to typeset a block of Java code, the command \Java{int␣i;} will typeset the code right inside the text, namely like this: int␣i;. Similarly, \C{int␣i;} will yield int␣i; while \Code{int␣i;} will yield int␣i; and \Pseudocode{int␣i;} will yield *int×i;*.

Internally, all of these command just call \lstinline. For instance, \Java⟨*text*⟩ is mainly a shorthand for

```
\lstinline[style=code, language=java, style=inline]⟨text⟩
```

Any options passed to command like \Java will be passed on to \lstinline. Note that the style=inline is used to setup different fonts for inline text. Do not mess with them. Finally, note that instead of curly braces you can also use a so called delimiting character, see the documentation of the LISTINGS package for more details.

Miscellaneous

There is a special style called escapemath that allows you to escape to math mode inside a listing using $. It is switched on by default for pseudo code, but not for normal code. It is useful for inserting meta-variables into a listing:

```
\begin{C++}[style=escapemath]
void $\langle \mathit{identifier}\rangle$ (int i)
{
  return i == 0 ? 1 : 1 + $\langle \mathit{identifier}\rangle$(i-1);
}
\end{C++}
```

yields

```
void ⟨identifier⟩ (int i)
{
  return i == 0 ? 1 : 1 + ⟨identifier⟩(i-1);
}
```

In inline code, spaces are shown by default. You can switch this off using the key showspaces=false as in \C[showspaces=false]{int␣i;}, which yields int i;.

2.11   Creating Figures and Tables

With the thesis class, you add figures and tables using the standard `figure` and `table` environments. You should always add a caption to a figure and the caption should be below it, while the caption of a table should be above it. You should always label the figure and you *must* reference all figures at least once in the text. See Question 3.31 on page 49 for some hints on what to write in captions.

Figures

The following code is typical code for creating a figure. See Figure 2.4 for the resulting typeset figure. Note that the font size is set to \small inside figure (and, also, inside tables).

```
\begin{figure}[htbp]
  \centering
  \textcolor{black!10}{\vrule width3cm height 3cm}
  \caption{Ode to Malewitsch.}
  \label{fig-ode1}
\end{figure}
```
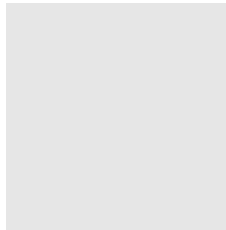

Figure 2.4: Ode to Malewitsch.

Tables

For tables, I recommend the following:

- Add horizontal lines around the table using the special command \uzlhline, see the example below, and possibly after the first line of the table. The exact size and color of the line will depend on the design, but you must insert it "by hand". Adding these lines will make tables visually consistent with listings and the rest of the thesis. However, you may also leave out the lines.
- Do not add any more or other lines. There is a saying in typography: "There is not a single table that cannot be improved by removing all lines." Lines are distracting and good spacing is a much better way of creating a visually pleasing table.
- Use the command \uzlemph with the text of column headers and (if needed) row headers. This command, which can also be used elsewhere, will typeset its argument in the same way as headlines are typeset (but in the usual font size). This will cause table heads to be visually consistent with the rest of the thesis design.

As an example, here is the code for Table 2.5 on the next page:

```
\begin{table}[htpb]
  \caption{Sounds made by different kinds of animals...}
  \label{fig-tab1}
  \centering
  \begin{tabular}{lp{5cm}}
    \uzlhline
```

```
    \uzlemph{Animal} & \uzlemph{Sound} \\ \uzlhline
    Cat & Meow \\
    Dog & Wuff or bark\\ \uzlhline
  \end{tabular}
\end{table}
```

Table 2.5: Sounds made by different kinds of animals, based on the reports by children of the age range 5 to 10. While the number *n* of children asked to report was small ($n = 1$) and the results are, thus, not fully representative of the general population, the author would like to point out that the ethics committee approved this survey.

Animal	Sound
Cat	Meow
Dog	Wuff or bark

2.12   Creating Graphics

External Graphics

Graphics (like plots, images, drawings or other data visualizations) can be added to a LATEX document in two ways: First, you can include an *external graphic* like a PDF file or a JPG file. Second, you can use "describe your graphic using LATEX commands". We discuss external graphics next, internal ones later on.

You include an external graphics using the \includegraphics command, which is a standard LATEX command. There is no need to include any packages for this, it is available automatically. For instance, you could say:

```
The university slogan \includegraphics{uzl-thesis-logo-slogan.pdf} in a
    sentence.
```

to get: "The university slogan IM FOCUS DAS LEBEN in a sentence."

As can be seen, the effect of the \includegraphics command is to directly include the graphic at the very position in the line where the command is used. Indeed, from TEX's point of view, an external graphic is indistinguishable from a black rectangle of the same size as the graphic.

You will rarely wish to put a graphic in the middle of a sentence (although there are applications). Instead, you will usually place it inside a {figure} environment: Recall that it is the job of the environment to creating "floating" text with a caption – and it is then the job of \includegraphics to replace the "text" by a picture. Here is an example of how you will usually do this:

```
\begin{figure}[htpb]
  \centering
  \includegraphics{uzl-thesis-logo-uzl.pdf}
  \caption{The logo of the University of Lübeck. It consists...}
```

```
    \label{fig-logo}
\end{figure}
```

The result is Figure 2.6.

Figure 2.6: The logo of the University of Lübeck. It consists of the university's seal together with the text "Universität zu Lübeck". The corporate design manual of the university requires this logo to be put at the upper left corner of title pages of university publications.

The \includegraphics command takes many options, the most important of which are likely height and width. These allow you to scale the graphic to a given height or width. *Avoid these options whenever possible.* The reason is that most graphics have a natural size (such as the logo) in which the text and fonts in the graphic are at the correct sizes. Any scaling will cause the graphic to become too large or too small. *Scaling is evil* and you will find more comments on this in Question 3.30 on page 49. All professors I know find scaled-down graphics with unintelligible text *among the most irritating things a student could possibly do.*

This means that when you *create* graphics with another program, make *sure* that any text in the external graphic has the same size as normal text in the thesis and that *no* scaling is needed.

Inline Graphics via TikZ

The alternative to external graphics are *internal* graphics. They are created using special LaTeX commands such as the following:

```
\tikz \draw [->] (0,0) -- (1cm, 2mm);
```

which yields ⟶ when used in a paragraph. A more complex example would be

```
\tikz [baseline, anchor=base] {
  \node [block = emph]                          (h) {Hello};
  \node [small node = emph blue, right = 5mm of h] (w) {Welt};
  \draw (h) edge[bend left=15, <->] (w);
}
```

which yields Hallo ⟷ Welt.

The Ti*k*Z package is used for these inline graphics and it is loaded automatically – so *if* you are going to create inline graphics, use Ti*k*Z. If you wish to learn Ti*k*Z, please read the tutorials from the manual (Tantau, 2019, Part I).

The thesis class sets up some styles for Ti*k*Z that you can either explicitly use or that are generally set up. For instance, the default arrow tip is setup according to the chosen design as well as the standard line width. You usually do not need to worry about these automatic settings.

Predefined Ti*k*Z Styles

There are several styles that are predefined and that you are "invited" to use:

base shape This is a style on which the styles described next are based. It is to be used with the \node command and will, for instance, fill the node with white color and will draw a thick border around it. The color that is used for the border can be passed as an argument, but see below for which colors you should use. Here is a simple example how this style is used:

```
\tikz \node [base shape, ellipse] {Hello};
```

yields (Hello).

The default color used for shapes is defined by the design (either black or the color ○zeangruen, which is the university's corporate design color, depending on the design).

small shape This style can be used in addition to base␣shape and will change the font to a smaller size and will reduce the inner seperation:

```
\tikz \node [base shape, small shape, ellipse] {Hello};
```

yields (Hello).

block A rectangular node: \tikz␣\node␣[block]␣{Hello}; yields │ Hello │.

small block A smaller version: Saying \tikz␣\node␣[small␣block]␣{Hello}; will now give │Hello│.

node A circular node, especially in a graph: \tikz␣\node␣[node]␣{n}; yields (n).

small node Small version: \tikz␣\node␣[small␣node]␣{n}; yields (n).

tiny node A small, unnamed circular node: \tikz␣\node␣[tiny␣node]␣{}; yields ○.

paper This is a special style that installs a "paper-like background" as fill color. I recommend using this style as a background for all cases where you wish to create the impression of showing an "original text" from another paper, that is, where you wish to show that the text or graphic "looks like this in the original". Consider for instance,

```
\begin{figure}[htpb]
  \centering
  \tikz \node [paper] {\includegraphics{uzl-thesis-logo-uzl.pdf}};
  \caption{The logo of the University of Lübeck, but now...}
  \label{fig-logo-paper}
\end{figure}
```

which yields Figure 2.7.

Figure 2.7: The logo of the University of Lübeck, but now with a "paper-like background". Note how, compared to Figure 2.6, a much stronger impression is created that this figure depicts something printed.

An example of how many of these styles can be used is shown in Figure 2.8 on the following page.

Predefined Colors

The thesis class defines a number of colors that you should use in graphics. You should *not* use colors like red or green: Pure green is a very light color and text in this color is hard to read on paper and impossible to read in an electronic document. Instead of pure green, a rather dark version of green must be used. In contrast, pure blue is already rather dark and only needs to be darkened very slightly. The following colors have been setup to provide a uniform contrast against a white background:

emph A red color. Used in an outline ▭ and filled ●
emph red This is the same as emph.
emph green Used in an outline ▭ and filled ●
emph blue Used in an outline ▭ and filled ●
emph black This is just black: Used in an outline ▭ and filled ●

Looking for more colors? Think carefully whether you really need more: It is hard to remember too many colors as a reader. It may be better to use a different style (like thicker lines) for purposes of further differentiation.

Designs for TikZ Graphics

In addition to the designs for the whole thesis, see Section 2.4, there are also three designs for graphics. Each of them redefines base␣shape, resulting in a different "look":

thesis outline shapes This TikZ style defines base␣shape and all styles built on top of it (like block and node) to a white background and simple thick line around the shape. For instance:

Merge sort visualization
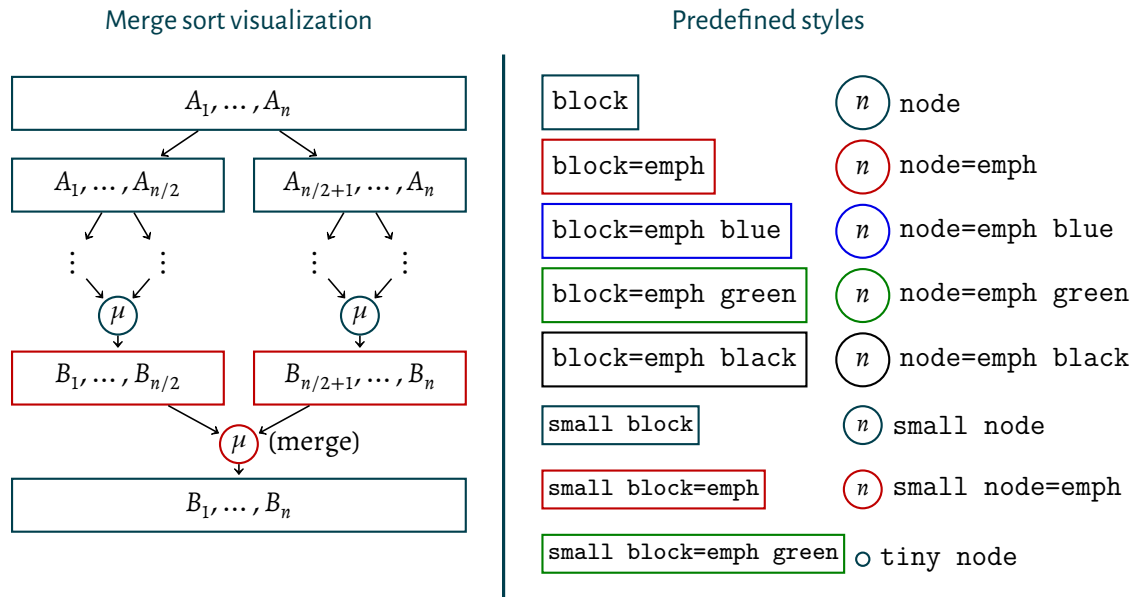
Predefined styles

Figure 2.8: An example visualization created with TikZ, see the template source for the code details. In the graphic, a number of predefined styles are used (like node or small␣block), each of which can be passed an optional color. These styles are setup automatically to produce visually pleasing shapes that go well with the overall layout and fonts.

```
\tikzset{thesis outline shapes} % generally set the design
\tikz \node [block] {Hello};
```

yields │ Hello │. See Figure 2.8 for a larger example.

thesis box shapes This style defines base␣shape similarly to the outline style, but fills the shapes with a light background:

```
\tikzset{thesis box shapes} % generally set the design
\tikz \node [block] {Hello};
```

yields │ Hello │. See Figure 2.9 on the following page for an example.

thesis flat shapes This style redefines base␣shape differently: The shapes are only filled and no border is drawn. This creates a stylish "flat" look:

```
\tikzset{thesis flat shapes} % generally set the design
\tikz \node [block] {Hello};
```

yields │ Hello │. See Figure 2.10 on the next page for a larger example.

Merge sort visualization

Predefined styles

$$A_1, \ldots, A_n$$

$$A_1, \ldots, A_{n/2} \qquad A_{n/2+1}, \ldots, A_n$$

$$\vdots \qquad \vdots \qquad \vdots \qquad \vdots$$

$$\mu \qquad \mu$$

$$B_1, \ldots, B_{n/2} \qquad B_{n/2+1}, \ldots, B_n$$

$$\mu \text{ (merge)}$$

$$B_1, \ldots, B_n$$

block	n node
block=emph	n node=emph
block=emph blue	n node=emph blue
block=emph green	n node=emph green
block=emph black	n node=emph black
small block	n small node
small block=emph	n small node=emph
small block=emph green	$\circ$ tiny node
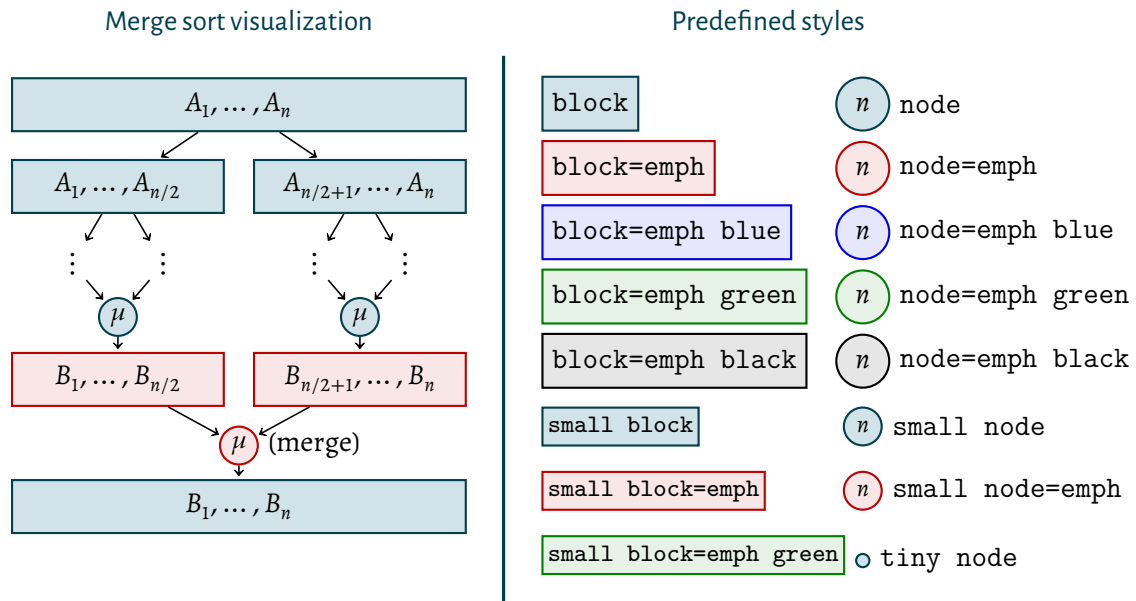
Figure 2.9: The same visualization as in Figure 2.8, but with the thesis␣box␣shapes option set. As can be seen, this option causes the predefined shapes to be filled. It is a matter of taste whether one prefers this over the outline style from Figure 2.8.

Merge sort visualization

Predefined styles

$$A_1, \ldots, A_n$$

$$A_1, \ldots, A_{n/2} \qquad A_{n/2+1}, \ldots, A_n$$

$$\vdots \qquad \vdots \qquad \vdots \qquad \vdots$$

$$\mu \qquad \mu$$

$$B_1, \ldots, B_{n/2} \qquad B_{n/2+1}, \ldots, B_n$$

$$\mu \text{ (merge)}$$

$$B_1, \ldots, B_n$$

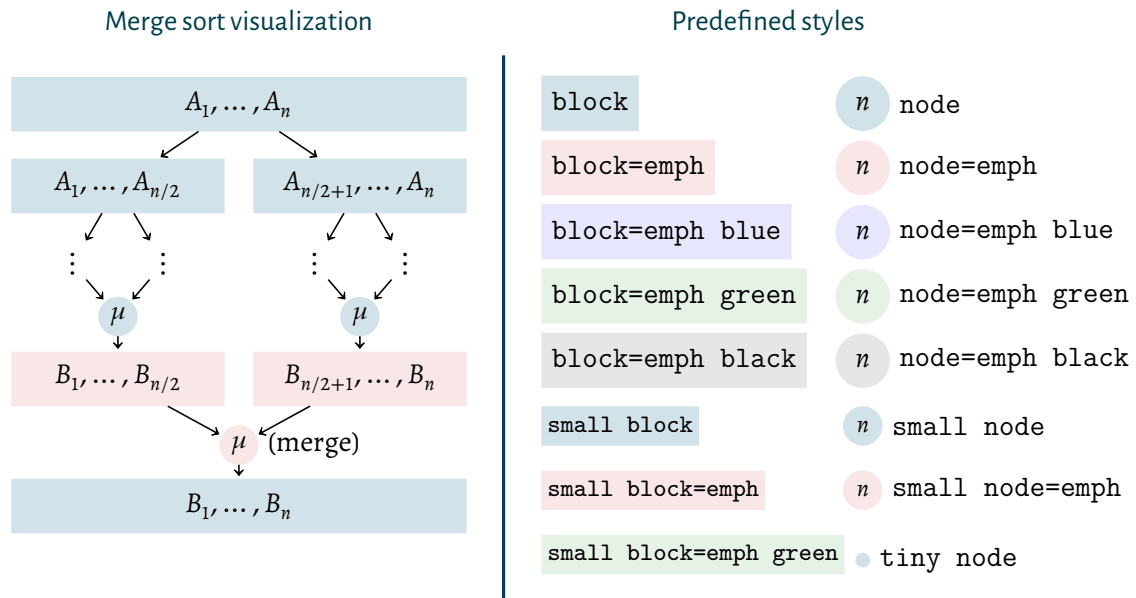|	
block	n node
block=emph	n node=emph
block=emph blue	n node=emph blue
block=emph green	n node=emph green
block=emph black	n node=emph black
small block	n small node
small block=emph	n small node=emph
small block=emph green	$\bullet$ tiny node

Figure 2.10: Once more, the same visualization as in Figure 2.8, but now with thesis␣flat␣ shapes option set. This option creates more "flat" shapes (without a border). Once more it is a matter of taste what one prefers.

2.13   Creating Cross References, Citations and the Bibliography

"We all stand on the should of giants" is a saying that refers to the fact that, in science, you never start from scratch: Your work always builds on that of many, many people before you. It is thus not only permissible, but actually required that you refer to the work of other people using *citations* – how this is done with the thesis class is explained below. In addition, apart from citing the work of other people, you will also often wish to point to different parts of your own thesis. The method of creating such *cross references* is also explained in this section.

Citations

In your thesis, you will and should use citations. This is done in LaTeX using the \cite command: If I wish to say that the TeX typesetting system was written by Donald Knuth, I could write

```
The \TeX\ system~\cite{Knuth1986} is due to Donald Knuth.
```

and would get "The TeX system (Knuth, 1986) is due to Donald Knuth." The \cite command looks up the *bibliography key* Knuth1986 in the list of bibliography entries (see below) and then inserts the citation into the text and the bibliography entry into the bibliography. All of this happens automatically, you just need to specify the entry in the list of bibliography entries and you need to add the citations at all points where you cite a book or an article in the main text.

The thesis class uses the BIBLATEX package for the bibliography handling (Kime and Lehman, 2018). You can use the different commands offered by the package, such as for instance \citeauthor to just cite the author names. There are a lot of commands, it is a powerful package.

Bibliography Entries (BibTeX Entries or Records)

In order to cite a book or an article, you say \cite{⟨*key*⟩} in the text, where ⟨*key*⟩ must be an entry in the list of bibliography entries. For historic reasons, these entries come in a weird format and must be processed using a special program called BibTeX.

I have tried to simplify the whole BibTeX business as much as possible for you:

– With the thesis class, you can place the bibliography entries right inside your (LaTeX) manuscript: At any point in the manuscript (I recommend somewhere near the end), you can use the {bibtex␣entries} environment. Inside this environment, you place all the bibliography entries using the BIBTEX format (if you are unsure about this format, look up the documentation of BIBLATEX, but the format is fairly self-explanatory). Listing 2.11 shows an example of how this works.

– You will then have to run lualatex on your manuscript and then run bibtex on the manuscript and then run lualatex once more. That should do the trick. (The effect of the {bibtex␣entries} environment is that when LaTeX is run on your manuscript, the text inside the environment gets written into a special file that is then processed by bibtex later on.)

Listing 2.11: Typical use of the {bibtex␣entries} environment inside a thesis manuscript.

```
\begin{document}
...
The \TeX\ system~\cite{Knuth1986} is due to Donald Knuth.
...
\begin{bibtex entries}
@Book{Knuth1986,
  author =        {Donald Erwin Knuth},
  title =         {The \TeX book},
  publisher =     {Addison-Wesley},
  year =          {1986},
}
...
\end{bibtex entries}
\end{document}
```

– Actually, the "right" program to use with BIBLATEX is not bibtex, but biber. However, the advantages of biber are only slight and setting it up correctly can be nontrivial. If you do wish to use biber, you must say

```
\UzLThesisSetup{biblatex={backend=biber}}
```

Do this only if you know what you are doing.
– If, for whatever reasons, you do not wish to use the {bibtex␣entries} environment, you can and must use the \addbibresource command of the BIBLATEX package. Do this only if you know what you are doing.

The Bibliography

The bibliography itself (the "list of references") is inserted automatically at the end of the thesis. Do not mess with this.

The bibliography style (the way the bibliography "looks") is also not up to you, it is dictated by university regulations. You *must* use the citation style of the journal *Cell* for all theses written in the area of the natural sciences (like Molecular Life Science) and you *can* also use so-called "numbered" citations for the all other subjects (but you can also use the citation style of *Cell* for them). The bottom line is that you must say in the preamble

```
\UzLThesisSetup{Alphabetische Bibliographie}
```

for a thesis in the area of the natural sciences and you can, alternatively, say

```
\UzLThesisSetup{Numerische Bibliographie}
```

for all other subjects. These commands will install the correct citation modes. Do not mess with them.[15]

---

[15] It really took me a *long* time to get them right.

Cross References

In scientific texts, you will often wish to reference an element of the text itself at some other place: For instance, in the text you wish to write something like "As Figure 3.2 shows, there is no evidence for..." where "Figure 3.2" is, of course, a figure in your thesis. In LaTeX, you use the commands \label and \ref for these purposes, see the LaTeX manual (Lamport, 1994) for more details, and the VARIOREF package is also automatically loaded. In particular, you can use the useful \vref command, see (Mittelbach, 2016) for the documentation.

3

Questions & Answers: How to Write a Thesis That Is a Pleasure to Read

Suppose that, like me (Professor Tantau), you are asked to grade a thesis that a hopeful student just turned in. Suppose that, like me, you have to grade a *lot* of theses (at least one per month) and you start reading it. Certainly, you strive to be as fair and even-minded as humanly possible when it comes to grading the thesis. In particular, you have the firm intention of determining your final grade based on the quality of the scientific contribution that the thesis makes.

As you start reading, you stumble over a number things. It starts with the table of contents that seems a bit confusing and after having read both the abstract and the one-page introduction you are still not quite sure what the author actually achieved and start to wonder "What is the point of all of this? And what did the guy actually *do* during the six months?" But you read on.

As you continue reading, your reading flow gets interrupted repeatedly by spelling mistakes, each of which makes you cringe a little. At first you think that the student might have introduced them during some last-minute revision work, but then you stop reading and start marking the spelling and grammar mistakes on five consecutive pages. You are somewhat disappointed when you find that there are about ten mistakes per page and the number does not appear to drop significantly as the text continues. By multiplying the number of mistakes per page with the number of pages, you extrapolate that the total number of mistakes must be in the range of 200 to 400 spelling mistakes. This makes you pretty sad and even a bit angry since you start to wonder "Why do I have to read this? Why couldn't the author be bothered to run a spellchecker on this text during the *six months* of working on it?" But you read on.

As you have a closer look at the main part of the text, you get determined to find out what author's main point actually is. It is a struggle since the author sometimes explains trivial stuff over two pages but then does not provide a single example with the main definition. Nor with any of the later results. But there are some tables, although it is a bit unclear what they actually show. Anyway, you are a professor and a professor can do anything, so you can also understand this text. It turns out that the trick is to quickly read the one paper that the student cites a zillion times – probably the only paper the student actually read. With the knowledge of that paper, you realize that the thesis actually contains

some nice ideas: perhaps not terribly deep results, but definitely nice ideas.

Here is the problem you face: How to grade a thesis that is literally littered with spelling mistakes, that is a mess concerning structure and exposition, that has a terrible bibliography, that forces you to read the original literature in order to understand what it is all about – and that contains really nice scientific ideas?

Welcome to my world. "Nice scientific ideas" can be graded in almost any way one likes: In a review, one can write that "the thesis certainly contains nice ideas, but utterly fails to demonstrate their actual relevance" or one can write "the thesis certainly contains nice ideas and these are likely to open new and interesting future approaches – a full demonstration of the practical relevance of the ideas is of course beyond the scope of a thesis". My point is that you, dear students, *do not want to infuriate the person who grades your thesis!* You want to turn in a thesis that is *a pleasure to read!* This will determine which of the two reviews will be written.

Of course, if your thesis proves P $\neq$ NP or presents a cure for cancer, you can turn in total gibberish, you can use a spelling that would make even your friends cringe when used in a chat, you can write your thesis on a napkin and the thesis can have two pages or 1000 pages – you will still get an outstanding grade. The other way round, no matter how stylish your thesis looks, how carefully it is phrased and how nicely you have wrapped the binding with leather – if you do not present solid scientific work in your thesis, you will still fail.

But in all other cases, you will just need to make an effort to create a well-written thesis. In the rest of this chapter, I try to give some hints on how to do this and will try to cover most of the famous *frequently asked questions* that I am, indeed, frequently asked. As with any recommendations, you can ignore them, but you do so at your own risk. But above all, remember that you should *always listen to your adviser!*

3.1   Q&A: Thesis Length

Question 3.1. *How many pages should my thesis have?*

Answer. There is no fixed length for a thesis, but for a bachelor's thesis the rule of thumb is 20 to 40 pages with 30 being ideal. For a master's thesis, aim for twice that, so 40 to 80 pages with 60 being ideal. The pages include everything following the table of contents (the pages numbered in Arabic in this class).

Reviewers do not appreciate theses that are too long or too short: If your thesis is much shorter than the suggested length, reviewers will wonder about the depth of the scientific contribution unless the presented results are absolutely brilliant. If your thesis is much longer than suggested, reviewers will get less and less inclined to continue reading: part of the difficulty of presenting scientific results is presenting them in a limited amount of words.

Question 3.2. *How long should the different parts of my thesis (abstract, introduction, main text, etc.) be?*

Answer. The length of the abstract must be around 150 words, independently of what kind of thesis you are writing. The length of the introduction should be around 10% of the

main text. Thus, if a bachelor's thesis has 30 pages, the introduction should have three to four pages (and not just one); a master's thesis can easily have an introduction with six to eight pages. The length of the conclusion should be around 5% to 10% of the main text. This leaves around 75% to 80% for the main text (because of the bibliography). You will find more information on the abstract, the introduction and the conclusion in Section 3.3 on the following page.

3.2   Q&A: Structuring a Thesis

Question 3.3. *Which parts should my thesis have? Is there anything that* has *to be there? Are there optional parts?*

Answer. Every thesis *must* follow the same basic structure:

1. Title (page),
2. abstract,
3. introduction,
4. main chapters,
5. conclusion and
6. bibliography.

A thesis that misses any of these parts is not really a thesis. The thesis class, by the way, makes it easy to generate this structure since you only need to make sure that the {document} environment contains a \chapter{Introduction} command and a corresponding \chapter{Conclusion} command – everything else will be inserted automatically at the right places and the class will give you an error message if anything is missing.

Question 3.4. *Which main chapters should my thesis have?*

Answer. While the introduction itself also has a fixed structure, see Question 3.9, there is no fixed structure for the main part. The number of chapters and sections really depends on the contents of your thesis. Usually, it is a good idea to have two to five main chapters (in addition to the introduction and the conclusion) and each can have anything between zero and ten sections. However, you should *never* have only a single section in a chapter. In such a case, remove the section command.

It is customary and useful to establish the structure of the thesis before you write the actual text. That is, use the \chapter and \section commands to create chapters and sections without any (or with only little) text, but with the tentative chapter and section titles already in place. Then create a more or less empty thesis and discuss the resulting table of contents with your adviser.

Question 3.5. *How important is the table of contents?*

Answer. The table of contents is one of the most important parts of your thesis. *Have a close look at it, repeatedly!*

The ideal table of contents tells you what happens in the thesis. It is tough to actually get this information into a table of contents, but this is what you should strive for: Organize the titles so that the really tell a story all by their own. As an example, consider the

table of contents of this document. Even without having read anything, a reader will see immediately see that this text has two parts or aspects: A part that obviously serves as some sort of "user's guide" on the thesis class and a part that contains a lot of "questions and answers" on how to write a thesis.

When you read the table of contents in isolation, you will also easily notice inconsistencies: For instance, it often happens that one section starts with an article (like "The Main Experiment") while another starts without one (like "Main Results"). Looking at the table of contents makes it easy to spot such inconsistencies (you should, of course, correct them). As an example, consider once more the table of contents of this document. It is, of course, no coincidence that *all* sections in the second part start with "Q&A". A less obvious non-coincidence is the fact that all sections of the first part start with the present participle of a verb and that some words get repeated: I adjusted the section names at a relatively late stage when writing this text to ensure this consistency – after I had had a look at a previous version of the table of contents.

3.3   Q&A: Abstract, Introduction and Conclusion

For any beginner, there is a very confusing aspect when it comes to *summarizing* the thesis: You have to do it three times! Once in the abstract, once in the introduction and once more in the conclusion. In the following, we have a look at why this is the case and how these three summaries differ.

Question 3.6. How long should the abstract be? I have heard it should be half a page?

Answer. The abstract should *always* be exactly one paragraph of 100 to 200 words (this is true for *every* abstract, even for a thick book).

Question 3.7. What is the purpose of the abstract?

Answer. The abstract summarizes and explains everything that happens in the thesis. Its job is to help someone decide "whether the thesis contains something that I am currently interested in" by just reading the abstract – this abstract is often shown for instance on web pages without showing the whole thesis. This means that if after reading the abstract someone thinks "sounds highly interesting, but I would have to read the thesis to know what, exactly, is shown," then the abstract is *bad* since it leaves the reader wondering. If after reading the abstract someone thinks "I do not need to read the thesis since it does not address the problem I am interest in," then the abstract is *good* (provided, of course, that the thesis really does not address the problem...).

This means that you must *name your results in the abstract.* Of course, you cannot (indeed, should not) go into any details, but an abstract should not include just a sentence like "In the thesis, new approaches are tried." Rather, write "In the thesis, three new approaches are tested: the foo-to-bar conversion, the bar-to-foo conversion and the bar-bar method. The evaluation shows that the first two outperform previous methods by up to 5% on average."

Question 3.8. And what is the job of the introduction?

Answer. The job of the introduction ("Einleitung" in German) is to introduce the reader to the subject, but also to *explain all results obtained in the thesis.* This means that, after having read the introduction, the reader should *know all major results obtained.* You may wonder what you should then write in the main text – the answer is, of course, that the main text has the job of "proving" that all the results you claim in the introduction actually hold.

Question 3.9. *Should the introduction have a special structure?*

Answer. It is customary to start the introduction with a motivation that explains what the subject is and why it is interesting. Put this right at the beginning, usually without a special section title like "Motivation". Just start explaining. Then, however, there should always be three sections, starting with "Contributions of this Thesis" or "Results of this Thesis" ("Beiträge dieser Arbeit" or "Ergebnisse dieser Arbeit"), then a section on "Related Work" ("Verwandte Arbeiten") and then "Structure of the Thesis" ("Aufbau dieser Arbeit"). Do not deviate from this structure (unless your adviser explicitly tells you to).

The "Contributions of this Thesis" section should explicitly state all of the main results – be aware that a very busy professor may actually skip the main text (sadly this does happen) and will *only* be aware of what you did from what is written here. Note that the contributions of a thesis are not always earth-shattering new fundamental breakthroughs. Indeed, they almost never are. Explain in clear words what you did and which scientific contributions you made. This may included:

- New results you obtained: This can include new mathematical statements, algorithms, experiment results, setups and more.
- Failed approaches: It is often scientific highly interesting to know that an approach does *not* work, so that other people do not need to repeat the approach once more.
- Literature work: It can be a scientifically nontrivial endeavor to read and then systematically present the relevant literature on a certain topic.

The "Related Work" section is the place where you point out what is already known about the subject and what other approaches have been tried. Here, you explain and cite other people's works. Do not be shy about this, even if you feel other approaches are better – that does in no way belittle *your* approach.

The "Structure of this Thesis" section should explain, usually in one paragraph, how your thesis is structured. There is no need to say things like "It starts with an introduction" or "The thesis concludes with a summary and an outlook" – we know and expect this. The interesting question is how the main part is structured. Try to make this text interesting and do not just list the titles of the sections: Why does a certain chapter come first? Can different parts be read independently? Where are the main results to be found?

Question 3.10. *Why do I have to write a conclusion? I have summarized the text already twice (in the abstract and in the introduction)! Summarizing it again makes no sense!*

Answer. You *always* include a conclusion in the thesis. It can be short (5% to 10% of the text), but it should be present. The job of the conclusion is to summarize the main text once more, but the difference to the previous two summaries is that the conclusion is addressed to someone who *has already read the thesis,* which allows you to go into much more

detail in the conclusion. Here you can write things like "The technically most difficult aspect was the proof of sub-case 4 of Lemma 3.1, where…" or "Especially the experiments IV and VIIa to VIIc proved difficult since…," which would make little sense earlier. The conclusion is the place to describe the highlights and problems encountered in the thesis "with hindsight" while the introduction needs to summarize things for someone not yet familiar with the subject.

Including an outlook is optional, if you do not include it, entitle the section just "Conclusion", otherwise "Conclusion and Outlook". The outlook should really be an outlook in the sense "where do we go from here." Do *not* use the outlook to list all the things that you wanted to do in your thesis, but did not find the time to do. In these cases, reviewers may get the impression that "the student seems to have been a bit lazy, considering the many things she or he did not manage to do." It is then better not to include an outlook.

3.4   Q&A: Writing Scientific Text

Writing the main part of your thesis, that is, writing the *actual text* can be daunting. Your thesis should be a "scientific text", but what does that mean, exactly? Surely, the text will be complicated since science is complicated – should you actually try to write a "complicated text"? Or should you strive to simplify things and explain your topic in layman's terms? In the following, we have a look at how "scientific text" is written (in particular, in Question 3.19, we have a look at why *we* have a look).

Question 3.11. *I have real trouble writing anything: I cannot even write a good first sentence for my text! How will I ever write 30 pages?!*

Answer. The first sentence is often the hardest sentence to write – so do not start with it! There is no need to start the text from the beginning with the perfect first sentence. Instead, start anywhere where you already know what you would like to say; the important lead sentences can come much later. This leads to an important trick: *Never stop writing at the end of a chapter or section, always write a few sentences for the next chapter or section before you call it a day.* Once you are in the flow, you will find it relatively easy to write text and also to write the starting sentences of a new chapter. When you continue writing on the next day or the next week, it will be much easier to just add something to text that is already there.

Question 3.12. *I still have trouble writing anything: I need hours to write a single good sentence!*

Answer. *Never* try to write "the perfect text". Good texts do not arise from good writing, but from good revising: You should constantly go over your text and revise it, reformulate things, remove parts and add new ones. At the beginning, a lot of "movement" happens and the text changes constantly, but after some time the text gets better and better and the changes less and less frequent. As a rule of thumb, *no single sentence remains unchanged from when it was first written to the sentence in the final version.*

Since you should and must revise everything anyway, it is not really important that you get the text right at the beginning. Thus, to overcoming the "writer's block", simply

write *something*, no matter how bad it is. Once *some* text is there, it is usually much easier to add more to it and to improve it.

Of course, this means that you will need *time* to go over the text repeatedly and to improve it. Do not assume that you can write a thesis in a week: While it might be physically possible to write down 30 pages in a week, it is *not* possible to do five revisions of these 30 pages in a week. The more time you have for revisions, the better.

Question 3.13.   *How should I start my first chapter?*

Answer.   Your whole thesis and each part of it can follow a simple pattern: Always start with a summary, followed by the details, followed by a wrap-up. On the top level, the title of the thesis actually summarizes the whole thesis! – and it is followed by the thesis text. This means, in particular, that the thesis title should be highly informative and should really tell us what is done in the thesis – do not try to be too funny or whimsical here. On the next level, the whole text starts with an abstract *and* an introduction, both of which summarize the text, followed by the main text, followed by the conclusion.

Now in answer to the question: You use the *same* pattern also for each chapter. This means that in each chapter the text before the first section is a summary of what happens in that chapter. This summary should not be a section-by-section account of what comes next, but rather an introduction to the chapters topic with an overview of what we should expect in the chapter.

You will hopefully no longer be surprised at this point to learn that each *section* in turn should also start with a short introduction and even summary of the section text: Try to make sure that the first paragraph already contains all the punchlines of the section.

You can actually even apply the pattern to each paragraph, if you want: The first sentence of a paragraph is the most important one and should lead the whole paragraph. This is also a helpful rule when you try to decide where a new paragraph should start: It starts when a new idea is started.

Question 3.14.   *Which words and concepts do I have to explain? Does the professor know all this stuff? Should anyone be able to read my text?*

Answer.   It is always difficult to decide "what do I need to explain, what does the reader already know?" Fortunately, there is a trick that works most of the time: Always address the text "to yourself, one year ago." That means, everything you already knew before you started working on the thesis does not need to be explained once more: for a thesis in computer science, there is no need to explain what "a programming language" is and what "xml" means; but when writing a thesis in which you extend a special library that you had never heard of prior to writing the thesis, you *do* need to explain what this library does. In a thesis on biology, you should not start explaining what a cell is or even what an amino acid is, but you should explain what the "parsimony assumption for the evolution of phylogenetic trees" is.

Question 3.15.   *OK, but what about the introduction and especially the motivation? Surely, there, I should start explaining things from the beginning?*

Answer.   Even at the very beginning in the motivation, you should not spend more than one sentence on stuff "that everyone knows". Starting with a very broad context is known as

the "Adam and Eve beginning" since it starts at the, well, beginning. Instead, you should (very) quickly come to the interesting scientific question you address in the thesis.

As an example, suppose you write a thesis on the question of how secure are speech recognition systems like Alexa or Siri. Then you should not start with something like "The first serious applications of artificial intelligence date back to the 1960's, but people have dreamed about intelligent machines for a very long time." This is a nice enough sentence, but has nothing to do with the topic of the thesis. A much better start would be "As more and more people use cloud-based speech recognition in their homes to control their environment, more and more people also give Internet companies access to potentially highly sensitive information." Note that while this sentence is still fairly easy to understand and not very technical (and thus appropriate for an introduction) it *is* about the subject of the thesis.

Question 3.16. *How do I explain what I did? Should I be modest? Or boastful?*

Answer. You may neither boast about your accomplishments nor should you be overly modest. It is unlikely that you reinvented all of science with your thesis, so do not try to create this impression. However, students also often are so unsure of their own achievements that the reader has trouble determining what the actual contribution of the author was!

For an example, suppose that for your thesis you were asked to find the longest nucleotide sequence that is common to the genome of all species in a large genome database (like the ENSEMBL database). Suppose that you found out that the maximal length was 200,123 base pairs. You should *not* write sentences like "This thesis demonstrates that evolution conserves huge quantities of genetic code over evolutionary timescales." (way too boastful and not really true) nor too-modest ones like "The maximal length of common genomes was studied." (What was studied, exactly? And who did it? What is a "common genome", by the way?). You need to write, clearly, what you did: "Since the genome of different species is related by evolution, all species share parts of their genome. In this thesis, I did a quantitative study of the amount of sharing by determining the longest primary sequence that is present in the genomes of all species from the public ENSEMBLE database. The longest sequence has a length of 200,123 bases pairs and is the coding sequence of a cell wall protein."[16]

Question 3.17. *I produced no new results! What do I do now?!*

Answer. Already "writing up things nicely that were already known" can be an important scientific contribution. Likewise, having done an in-depth literature research is important for science. Naturally, if you have made some discoveries, explain them in the introduction and in the main text, but science is as much about known things as it is about new things.

By the way, it is also important to point out things that did *not* work out. Knowing that a certain approach does not work can be a more important scientific contribution than knowing that something gives a small improvement over previous methods: If you

---

[16] I made all of this up! I do *not* wish to read something like "As pointed out by (Tantau, 2019), all species share 200,123 base pairs" in future papers...

have shown that an approach does not work, other people will not waste time trying this approach again.

Question 3.18. *How do I write "scientifically"? Should I use complex sentences to describe complex topics and simple sentences for simple topics?*

Answer. A good "scientific" text is a text that is easy to understand – not a text that uses grand words and involved phrase constructions. Indeed, *especially* when a subject is hard to understand, it is *particularly* important to describe it in a *simple* way. This does not mean that one must *simplify* things, but just that the description should be clear and easy to follow – and, spoiler alert, a simple sentence structure and simple words are easier to understand that a convoluted structure with much technical jargon.

As an example, suppose you wish to explain the concept of public key cryptography. A seemingly very scientific – but in fact just bad – explanation might be "Public key cryptography is based on private–public key pairs (k_e, k_d) that encrypt and decrypt plain and cover text (and *vice versa*) using cryptographic hash functions with back-doors." While perfectly true, this convoluted sentence sheds almost no light on the question of what public key cryptography is or does or how it works. Do not put such sentences into your thesis.

What you should do instead is to use plain words in general and the scientific words where needed and where they are appropriate. Likewise, you should split up sentences when they become too long. Here is a much better explanation of public key cryptography: "Public key cryptography allows two persons who have never met to exchange messages over a public network in the presence of an eavesdropper. It is based on mathematically transforming messages by first *encrypting* them with a *public* key and then later on *decrypting* them with a *private* key."

Here are some further some simple hints:

– Simple is better than complicated. When in doubt, use the simpler word or simpler structure: Instead of "the addition of the elements in the set M yield 10" write "the sum of M is 10". Instead of "Assuming that A is an influencing factor for how large the standard deviation of B is, we may conclude that the correlation between A and B is actually based on a causal effect of A on B" write "If A affects how strongly B varies, A causes B."

  Especially in German text, avoid too complicated sentences.
– Vary the sentence length. As the previous sentence shows, short sentence can have a lot of "punch" since the reader can read and understand them at a glance, while longer sentences – like the present one – allow you to present longer thoughts. Again, if in doubt, try a short sentence: Instead of "The very last finding in the just-presented analysis rather clearly shows that one can no longer support the assumption that the previous findings are correct" write "This is wrong."
– Use short, simple words, when possible. This is especially true in German texts. This does not mean that you should replace a scientific term by something seemingly simpler (it would be utterly wrong to replace "nondeterministic" by "random") but that no complex words are introduced where simple ones would do. Instead of "The possibility exists to deterministically simulate the totality of the computational branches

done by nondeterministic general Turing machines" write "Nondeterministic Turing machines can be simulated by deterministic ones."

–   Read a book on how to write scientific text, like then one by Alley (1996).

Question 3.19. *In scientific texts, no one uses "I", it is always "we". Should I also do this? (Should we also do this?)*

Answer. The question "Should we also do this?" already shows that it is ridiculous to always use "we" when "I" is actually meant. No, you should not always use "we". Rather, the basic rules are as follows:

–   In a single-author paper (such as your thesis), the word "we" always refers to "the reader and me" (that is, "you and me"). Thus, you write "we will see later on" since you mean "you (and I) will see later on" and certainly not "I will see later on".

–   Since "we" means "you and me", you cannot use "we" when the reader does not feel addressed. You cannot write "We think that this is a good approach" since you force an opinion on the reader (who may feel that this is a terrible approach). Do not use phrases like "we think", "we believe", "we hope" and so on. You also cannot write "we did the following three experiments" since you did the experiments, but not the reader.

–   In all of the cases where you cannot write "we", I recommend writing "I" instead. "I believe that this is a good approach" is a clear statement with which one may or may not agree, but it is correct and perhaps even bold. "I did the following three experiments" is a simple factual statement and there is no reason not to write it.

    Some students (and even professors) fear that too many "I" in the text make the text look narcissistic and there is a certain truth to this. Try to use "I" mainly in the part "Contributions of this Thesis", where it is your job anyway to make clear what *you* did. In other parts, you can often rephrase things slightly: Instead of "I did three experiments to test whether pigs can fly" you can also write "for this thesis, three experiments were done to test whether pigs can fly" or "pigs cannot fly according to the three experiments done in the context of this thesis" or "the three pig-flight experiments of this thesis show that pigs do not fly" or one of many other variations.

    However, I recommend that you do *not* write "The author of this thesis has done three experiments" since addressing yourself as a third person is bad style in my opinion. Also, I think it would be really cool to read "I did three experiments to test whether pigs can fly" in a future thesis...[17]

3.5   Q&A: How to Write Mathematical Text

Writing mathematical text is, unfortunately, as tricky as writing normal text: There is the mathematical *language*, the way mathematicians phrase things, but there is also the problem of typesetting mathematical text and of getting all the symbols right. Mastering both – the phrasing and the typesetting – is not easy and cannot be fully explained in a few paragraphs. But I can, as always, give you some hints to get you started.

---

[17] Assuming the ethics committee approved the experiment.

Question 3.20. *For mathematical text, should I use as many formulas and symbols as possible? Or should I perhaps try to phrase everything while using as little symbols as possible? Does this matter?*

Answer. There is no "right" amount of mathematical formalism: It really depends on what you describe. Beginners tend to use too much formalism and tend to write things like "Assuming that $A \cap B = \emptyset$, then..." instead of "If A and B are disjoint, then...", but sometimes lengthy explanations are much easier to phrase using symbols: Instead of "Let x be the limit for n going to infinity of the series whose nth element is $e^{-n}n^2$" write "Let $x = \lim_{n \to \infty} e^{-n}n^2$". The only general rules you should be aware of are:

- Every text in mathematical mode (anything surrounded by \$'s in TeX) should be a self-contained formula. Never write "let n be $x+$ two divided by 4". Write "let $n = (x+2)/4$" or "let n be $(x+2)/4$" or "let n be x plus 2 and then divided by 4" (although this is too lengthy).
- Every mathematical text *must* be in mathematical mode. For instance, do not write "Let␣n␣be␣an␣integer.", but "Let␣\$n\$␣be␣an␣integer.".
- Read the book „*Das ist o. B. d. A. trivial!": Tipps und Tricks zur Formulierung mathematischer Gedanken (Mathematik für Studienanfänger)* by Beutelspacher (2009).

Question 3.21. *There are so many environments in TeX for mathematical text! Which one should I use?*

Answer. There are many possible environments for displaying mathematical text (like `{array}`, `{align}`, `{equation}` and more) on a separate line. I recommend always using `{align*}` and to use `{align}` (the numbered version) when a numbering is really essential. Use other environments only when `{align}` lacks a specific feature. Most of the time, however, you should just inline the mathematical text using \$'s.

Question 3.22. *Which font should I use for my special mathematical object?*

Answer. The rule is that *anything variable* should be in italics, while everything *constant and fixed* should be upright. For instance, in "$\sin^2 x + \cos^2 x = 1$" the text "sin" is upright since there is "only one sine function" and also the numbers 2 and 1 as well as the plus sign and the equality sign are all upright. In contrast, x is in italics since this variable can represent an arbitrary (variable) number.

Most of the time, TeX will get these things right automatically, but there are cases where you may need to help:

- A simple uppercase letter is interpreted as a variable, which is not correct, when the letter is actually part of a name. For instance, if you write \$P␣=␣NP\$ then TeX thinks "the user wants me to typeset that the variable P equals N multiplied by P; so I will typeset P and N in italics and maybe add a bit of space between N and P", resulting in "$P = NP$". What was actually meant was "the complexity class P equals the complexity class NP" and should have looked like this: "$\mathrm{P} = \mathrm{NP}$". The trick is to use the `\mathrm` command, which tells TeX to typeset something in an upright font:

```
$\mathrm{P} = \mathrm{NP}$
```

- In addition to \mathrm there is also \mathit, which is needed for longer variable names in some fonts: Suppose you have a variable called "fit". Now, with the Computer Modern fonts, just writing $fit=1$ makes T_EX think "the users wants me to typeset f times i times t equals 1, so I better add a bit of space between all the letters". However, this is just *one* name, so we must actually write $\mathit{fit}=1$ to get "$fit = 1$".

3.6   Q&A: Language and Spelling

Question 3.23. *Should I write my thesis in English or in German?*

Answer. If you are a German speaker, write your thesis in English *only* if your English is pretty good. Students sometimes tell me they want to write their thesis in English "in order to practice". While this seems like a good reason, it is actually a bad idea: Remember the introductory story from the beginning of the chapter? A thesis written in very bad English is a pain to read – and you do not wish to pain the person who grades it.

There is no reason to write a thesis in English unless you have a good command of the language. If you wish to learn English, go take a course, take a vacation in London or write a shorter paper together with someone who *is* good at English.

(You should, of course, always try to improve your English whenever possible since it is the language of science. But you should *not* try to turn your adviser into a language coach.)

Question 3.24. *How do I avoid spelling mistakes?*

Answer. Well, use a spell checker. Quite frankly, when I get a thesis with several spelling mistakes that any spell checker would have caught, my pulse actually rises as I think "why the ⟨beep⟩ do I have to read this ⟨beep⟩ where the author could not be ⟨beep⟩ bothered to run a spell checker just once?" You will *really* annoy your reviewers with obvious mistakes.

However, in addition to the mistakes that any spell checker will catch, there are of course many grammatical mistakes that one can make. In English texts, the problem is typically the word order that Germans get wrong; in German texts, students like to add and remove commas at random places throughout the text. Well, there a rules for all of this. If you could not be bothered in school to learn them, learn them now – there is no time like the present. You *must* follow them.

Of course, most of the time students do not make grammatical mistakes because they simply do not know where the comma should go, but because they just make a mistake and do not catch it later on. The trick is to *read, re-read and re-re-read* the text as often as necessary until almost all mistakes are gone.

It helps to read the text slowly.

It helps to have someone else read the text.

It helps to read the text backwards (yes, really: you cannot read quickly in this case).

Whatever you do, a well-written text should not have more than one spelling mistake every few *pages* – and not one every few *lines* let alone one every few *words*.

Question 3.25. *In English titles most words seem to start with an uppercase letter. Is this normal?*

Answer. In English, it is customary to spell (almost) all words with uppercase letters in titles including chapter and section titles. So, the title of the first section of this text is "Contributions of this Thesis" rather than "Contributions of this thesis". However, as can be seen, some words like "of" or "this" are not capitalized: Usually, conjunctions and prepositions are not capitalized. To a certain degree, you can choose, how many words you capitalize: some people capitalize everything with the sole exceptions of "a", "an" and "the", other people do not to capitalize at all, but this is unusual. Whatever "amount of capitalization" you choose, stick with it! It is plain *wrong* to write "Contributions of this Thesis" and then "Structure of This Thesis".

Question 3.26. *Do I have to also capitalize words in titles in German?*

Answer. In German you have to capitalize according to the German spelling rules. So, you would write "Untersuchungen zum genetischen Code" and not "Untersuchungen zum Genetischen Code", which would be plain wrong – even in the title of the thesis.

Question 3.27. *Which quotation marks should I use? And how do I enter them?*

Answer. Getting quotation marks right used to be somewhat complex in LaTeX; these days the main problem is to know which one are the right ones:

Language	Example	"Opening Form"	"Closing Form"
American English	"Example"	66	99
British English	'Example'	6	9
German first alternative	„Beispiel"	99	66
German second alternative	»Beispiel«	»	«

Anything else is just wrong.

I strongly recommend that you find out how you can directly enter the Unicode characters for the quotation marks of your choice and directly enter them in the text.

3.7   Q&A: Creating Figures and Tables

Question 3.28. *How many figures should my thesis have?*

Answer. I once knew a person who complained to me that in her PhD thesis on analytic *geometry* she did not have a single graphic. The reason was that all of her thesis was about infinite dimensional spaces and these are a bit hard to visualize in two dimensions.

A thesis can have anything between zero and dozens of figures. However, make sure that "each figure counts": Readers pay special attention to figures and if they do not understand them or cannot read them clearly, they will be much more annoyed than by a poorly phrased paragraph. It also takes rather a lot of time and energy to create a good figure, so think twice before adding a figure. Finally, too many figures tend to create the impression that one is not reading a thesis, but a manual (or, worse, is looking at a picture book): If you have a lot of figures, it may be better to leave some of them out and to concentrate on the really important ones that best convey the main message.

Question 3.29. *What program should I use for creating graphics?*

Answer. I recommend not to use a separate program, but to use TikZ and to create the graphics right inside the LaTeX manuscript, see Section 2.12 on page 28. But this recommendation may be slightly biased (I programmed TikZ...).

Question 3.30. *Thanks for TikZ, but I still wish to include graphics that I produce with my favorite drawing program. Surely, this is no problem?*

Answer. Technically, it is no problem to include PDF, JPG or PNG files, see Section 2.12 on page 27. However, there is one *big danger: Scaling!* Scaling can be evil for two reasons:

– Scaling changes not only the size of a graphic, but also the line widths. If you scale down a vector graphic by a factor of ten to fit it into your thesis, all lines will be so thin that they are almost invisible. The other way round, if you magnify a graphic using scaling, the lines may become so thick that they look almost comical.
– Scaling does even worse things with font sizes. The correct font size for use in figures in a thesis is 10pt. Good luck getting this font size correct when including an external graphic – text in graphics will almost always be way too small or way too large. Unintelligibly small text *really* annoys reviewers and we see it all the time in tables. Too big text makes the thesis look like a child's book. You do not want your thesis to look like a child's book.

Apart from making it hard to get the font sizes right, getting the fonts *themselves* right is even harder. Good luck creating a graphic with the correct mathematical symbols from the Alegrya font used in it.

Ideally, only include external pictures and create graphics with text in them using TikZ. If this is not an option, try to setup things so that no scaling is done and that the font size is 10pt and the normal line width is 0.4pt to 0.6pt.

Question 3.31. *How long should captions (for figures, tables, graphics, etc.) be?*

Answer. I recommend using *very long captions* (up to five lines) with figures and tables. The reason is that people look at figures and tables before they look at the main text. Figures should be "understandable all by themselves" and the caption has the job of ensuring this. In other words, make figures "as independent of the main text as possible." Naturally, you should still reference all figures in the text; *never* add a figure that is not mentioned in the text.

The caption of Figure 2.4 on page 26 demonstrates the problem with a short caption: Most readers will have no idea what this is about. A much better caption is the one of Figure 3.1 on the following page.

3.8   Q&A: How to Cite and What to Cite

Several well-known politicians recently lost their jobs because of some missing or incorrect citations in their PhD theses. Clearly, getting these citations right must be something important. Well, there is a very simple basic rule that the politicians violated: For
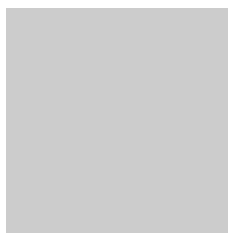
Figure 3.1: The depicted square is intended as an "ode" (that is, as a praising and – in this case – slightly ironic reference) to the famous painting *Black Square* by Kasimir Malewitsch from 1915, which depicts a black square. The above square is clearly not black, but just gray, and is hence intended as a whimsical reference.

every word of the text, it must be *crystal-clear who had the idea for that word.* There is nothing wrong, at all, with citing even a whole page from another paper – as long as it is clear that this is text from another paper!

Question 3.32. *When and where do I have to add citations?*

Answer. In accordance with the basic rule ("for each word of the thesis it must be crystal-clear who had the idea for that word") whenever an idea or a concept is from someone else, a citation is added. You add these citations next to the point of the idea or the text using the \cite command or related commands such as for instance \citeauthor, see the (Kime and Lehman, 2018) manual for more details.

Question 3.33. *Where can I get bibliography records (the BibTEX entries)?*

Answer. Many servers allow you to download BibTEX records for books and articles, but *be warned that these very often have very poor quality.* You *must* go over them by hand and often you will have to correct them. Ask your adviser if you are unsure.

Question 3.34. *How many entries should there be in the bibliography?*

Answer. There are no lower or upper bounds, but reviewers use the entries in the bibliography[18] as a measure of how much literature research has been done. If there are only one or two (or even zero) scientific articles, one old textbook and a lot of web pages that are cited, I would seriously doubt that a lot (or, for that matter, *any*) literature research has been done. On the other hand, if there is an endless list of entries in the bibliography, only very seldom did the student actually read all of these papers – more likely, most entries in the list have just been copied from somewhere else without the student's actually having looked into them.

The bottom line is that you should do a solid literature research (you should spend *at least* two *weeks* on that!) and then cite the five to fifteen papers and books that are really relevant and that you have actually read.

---

[18] Depending on the design used to typeset this text, the word *bibliography* gets hyphenated here in a very strange way: bib-li-og-ra-phy. It turns out, this is the correct English hyphenation, which I find weird since the word is clear composed of *biblio* (book) and *graphy* (writing). Luckily, TEX's hyphenation rules are pretty good and get this right.

Question 3.35. *Do I have to read all papers that I cite?*

Answer. Yes, you do – perhaps not completely, but you must read the part of the text to which you refer in your citation. Resist the temptation to just copy citations from other papers.

Question 3.36. *Can I cite Wikipedia?*

Answer. Yes, you can, but only if there is not a better scientific source available. For example, if you wish to cite the notion of a "Turing machine", you should not point the reader to the corresponding article in Wikipedia, but to an article in a standard textbook on theoretical computer science.

4

Conclusion

This template document got much longer than I had initially intended with more and more hints and comments becoming part of the text. The reason is, of course, that writing a thesis is not easy since there are a *lot* of things to consider. However, you have six months to write your thesis, so you stand a decent chance to get most things right.

Do some great scientific research now and report on it in a thesis that is a pleasure to read.

Bibliography

Alley, M. (1996). The Craft of Scientific Writing. Third Edition. Springer.

Arseneau, D. (Sept. 16, 2013). *The URL package. Verbatim with URL-sensitive line breaks*. Version 3.4. URL: http://www.ctan.org/pkg/url (visited on 06/11/2019).

Beutelspacher, A. (2009). „Das ist o. B. d. A. trivial!": Tipps und Tricks zur Formulierung mathematischer Gedanken (Mathematik für Studienanfänger). Ninth, updated edition. Vieweg+Teubner Verlag. DOI: 10.1007/978-3-8348-9075-7.

Braams, J.L. and Bezos López, J. (June 3, 2019). *The BABEL package. Multilingual support for Plain TeX or LaTeX*. Version 3.32. URL: http://www.ctan.org/pkg/babel (visited on 06/11/2019).

Downey, R.G. and Fellows, M.R. (2013). Fundamentals of Parameterized Complexity. Texts in Computer Science. Springer. DOI: 10.1007/978-1-4471-5559-1.

Kernighan, B. (1974). *Programming in C – A Tutorial*. Tech. rep. Bell Laboratories.

Kime, P. and Lehman, P. (Oct. 30, 2018). *The BIBLATEX package. Sophisticated Bibliographies in LaTeX*. Version 3.12. URL: https://github.com/plk/biblatex (visited on 06/11/2019).

Knuth, D.E. (1986). The TeXbook. Addison-Wesley.

Lamport, L. (1994). LaTeX: A Document Preparation System. Second edition. Addison-Wesley.

Mittelbach, F. (Feb. 16, 2016). *The VARIOREF package. Intelligent page references*. Version 1.5c. URL: http://www.ctan.org/pkg/varioref (visited on 06/11/2019).

Rahtz, S. and Oberdiek, H. (Nov. 30, 2018). *The HYPERREF package. Extensive support for hypertext in LaTeX*. Version 6.88e. URL: https://github.com/ho-tex/hyperref (visited on 06/11/2019).

Tantau, T. (2019). *The TikZ and PGF Packages: Manual for version 3.1.3*. URL: https://github.com/pgf-tikz/pgf.

The LaTeX Team (Sept. 2, 2017). *The AMSMATH package. AMS mathematical facilities for LaTeX*. Version 2.17a. URL: http://www.ams.org/tex/amslatex.html (visited on 06/11/2019).