Die folgenden Aufgaben haben alle mehr oder weniger viel mit der Vorlesung zu tun … Sie sollen Ihnen aber die Gelegenheit bieten, sich – falls Ihnen über die Weihnachtstage langweilig werden sollte – ein wenig mehr mit Ihrem neuen Lieblingstextsatzsystem, zu beschäftigen.

Alle Aufgaben sind Bonusaufgaben – dafür sind sie teilweise recht anspruchsvoll. Punkte werden vor allem für besonders kreative Lösungen und Ansätze vergeben. Sollten Sie über Weihnachten lieber Zeit mit Ihrer Familie oder Ihrem Kater verbringen, verpassen Sie auch nichts, wenn Sie die Aufgaben nicht bearbeiten.

★ Bonusaufgabe W.1: TeXnische Weihnachtsdekoration

+4 Bonuspunkte

a) Zeigen Sie Ihre Zeichenkünste und erstellen Sie ein – möglichst schönes und TeXnisch anspruchsvolles – weihnachtliches Motiv mit Ti*k*Z. Ihrer Kreativität sind keine Grenzen gesetzt.

b) Wenn Sie glauben Ihr Bild könnte es mit denen auf TeXample.net aufnehmen, schicken Sie den Sourcecode ruhig an den Betreiber, Stefan Kottwitz, mit dem Vorschlag, auch Ihr Beispiel zu übernehmen.

c) Alternativ können Sie versuchen, Ihrem Bild Leben einzuhauchen: Mit dem Paket `animate` lassen sich sich einfache Animationen aus einzelnen Frames erstellen.[*]

Falls sich Ihr Motiv dazu eignet, erstellen Sie mehrere Versionen, die sich leicht unterscheiden und fügen diese später zu einer Animation zusammen.[†]

Abgabe: Quellcode und fertiges Dokument ausgedruckt, Quellcode per Mail.

★ Bonusaufgabe W.2: Weihnachts-Rätselheft

+4 Bonuspunkte

Ein beliebter Ferienspaß ist das Lösen von Kreuzworträtseln, Sudokus und sonstigen Spielchen. Überraschen Sie Ihre Lieben doch mal mit einem selbst geTeXten Rätselheft.

Verschaffen Sie sich einen Überblick, welche Pakete es auf CTAN gibt, die für den Satz solcher Rätsel geeignet sind und erstellen Sie damit ein kleines Rätselheft, das einem eine Weile die Zeit vertreiben kann.

Abgabe: Quellcode und fertiges Dokument ausgedruckt, Quellcode per Mail.

★ Bonusaufgabe W.3: TeX-Adventskalender

+4 Bonuspunkte

Erstellen Sie einen Adventskalender – also ein Dokument, das, je nach dem an welchem Datum es kompiliert wird, etwas anderes beinhaltet. Um zu unterscheiden welcher Tag ist können Sie zum Beispiel LuaLaTeX verwenden und mit `\directlua{`⟨*lua-Code*⟩`}` beliebigen lua-Code ausführen, oder das Paket `ifthen` zu Hilfe nehmen, mit dem man if-Statements in LaTeX erzeugen kann.

Alternativ bietet sich auch das etwas modernere Paket `etoolbox` an, das if-Statements zur Verfügung stellt.[‡]

Abgabe: Quellcode ausgedruckt und per Mail.

[*] Leider werden diese Animationen nicht in jedem PDF-Viewer korrekt angezeigt.

[†] Ein PDF mit den genau den Dimensionen des Inhalts (also ohne Papierformat und ohne Rand) können Sie mit der Dokumentenklasse `standalone` erstellen.

[‡] Suchen Sie hierzu in der Dokumentation nach „toggle“ bzw. „tests“.

Lösung W.3

Mit den TeX-Primitiven `\day` und `\month` lässt sich direkt auf Tag und Monat des aktuellen Datums zugreifen. Das Paket `ifthen` bietet praktische Makros für die Arbeit mit if-Conditions. Damit lässt sich zum Beispiel abfragen, ob gerade Dezember ist:

```
\ifthenelse{\month = 12}{Es ist Dezember!}{Im Moment ist \emph{nicht} Dezember.}
```

Das erste Argument von `\ifthenelse` stellt dabei eine Bedingung dar, die entweder wahr oder falsch sein kann. Ist die Bedingung wahr, so wird der Inhalt des zweiten, ist sie falsch, der Inhalt des dritten Arguments ausgegeben. Aus solchen `\ifthenelse`-Konstruktionen lässt sich jetzt leicht z. B. ein Adventskalender erstellen.

★ Bonusaufgabe W.4: Quine

+6 Bonuspunkte

Eine besondere Freude für die Programmiererin und den Programmierer sind Programme, die ihren eigenen Code ausgeben, ohne dabei eine Eingabe zu benötigen. – sogenannte Quines. Probieren Sie mit TeX einen Quine zu erzeugen! Die TeX-Datei soll also im pdf genau den Code ausgeben, mit dem sie selbst geschrieben ist. Es sollen keine zusätzlichen Befehle im Quellcode stehen, die nicht im pdf als Ausgabe erscheinen!

Da TeX die Möglichkeit hat, auf seinen eigenen Sourcecode zuzugreifen lässt sich diese Aufgabe genaugenommen relativ einfach lösen. Eine solche Lösung stellt zwar auch eine gute Übung dar, als richtiger Quine zählt aber nur, was ohne diese Funktion auskommt.
Abgabe: Quellcode / fertiges Dokument ausgedruckt, Quellcode per Mail.

Lösung W.4

Auch wenn TeX theoretisch auf den eigenen Quellcode zugreifen könnte um diesen einfach als verbatim-input einzulesen, wäre das genau genommen kein echter Quine. Um einen Quine zu erstellen müssen wir Makros definieren, die den gewünschten Code enthalten und die wir sowohl zum Ausführen des Codes als auch für die Ausgabe desselben aufrufen können.[§]

In plainTeX könnte ein Quine beispielsweise so aussehen:[¶]

```
\tt\obeylines
\nopagenumbers\let~\string
\def\a{~\tt~\obeylines
~\nopagenumbers~\let~~~\string
~\def~\a~{\b~}
~\def~\b~{~{~\def~~~{~\string~~~\string~}~\def~\b~{~\string~\b~}~\a~}~}
~\a~\b~ye}
\def\b{{\def~{\string~\string}\def\b{\string\b}\a}}
\a\bye
```

[§] Eine relativ detaillierte Beschreibung von Quines (und etwas theoretischen Hintergrund) findet man auf der Seite. http://www.madore.org/~david/computers/quine.html

[¶] Quelle: http://tex.stackexchange.com/a/93930

Eine in LaTeX implementierte Alternative wäre zum Beispiel:¶

```
\documentclass{article}
\usepackage[T1]{fontenc}
\begin{document}
\obeylines\thispagestyle{empty}
\let~\textbackslash
\newcommand{\asciitilde}{\raisebox{-.5\height}{\textasciitilde}}
\newcommand{\mymyself}{{\let~\asciitilde
\renewcommand{\{}{\textbackslash\textbraceleft}
\renewcommand{\}}{\textbackslash\textbraceright}
\renewcommand{\asciitilde}{\textbackslash asciitilde}
\renewcommand{\mymyself}{\textbackslash mymyself}\myself}}
\newcommand{\myself}{
~documentclass\{article\}
~usepackage[T1]\{fontenc\}
~begin\{document\}
~obeylines~thispagestyle\{empty\}
~let\asciitilde~textbackslash
~newcommand\{~asciitilde\}\{~raisebox\{-.5~height\}\{~textasciitilde\}\}
~newcommand\{~mymyself\}\{\{~let\asciitilde~asciitilde
~renewcommand\{~\{\}\{~textbackslash~textbraceleft\}
~renewcommand\{~\}\}\{~textbackslash~textbraceright\}
~renewcommand\{~asciitilde\}\{~textbackslash asciitilde\}
~renewcommand\{~mymyself\}\{~textbackslash mymyself\}~myself\}\}
~newcommand\{~myself\}\{\mymyself\}
~myself
~end\{document\}}
\myself
\end{document}
```