Ne příliš stručný úvod do systému $\LaTeX\,2_\varepsilon$

Neboli LaTeX2e v 73 minutách

Tobias Oetiker

Hubert Partl, Irene Hyna a Elisabeth Schlegl

Michal Kočer, Pavel Sýkora

Verze 2.2-Beta, 25 leden 1996
Verze překladu CZ-0.9-Beta, prosinec 1998

Poděkování

Většina materiálu v této příručce užívá původního rakouského dokumentu *Úvod do LaTeXu 2.09,* jehož autory jsou:

Hubert Partl <partl@mail.boku.ac.at>
Zentraler Informatikdienst der Universität für Bodenkultur Wien

Irene Hyna <Irene.Hyna@bmwf.ac.at>
Bundesministerium für Wissenschaft und Forschung Wien

Elisabeth Schlegl <no email>
in Graz

Zajímá-li vás původní německy psaná verze, upravená Jörgem Knappenem pro LaTeX 2_ε, zkuste se podívat na libovolné zrcadlo archivu CTAN do adresáře `CTAN:/tex-archive/info/lkurz`. V České republice je archiv CTAN zrcadlen na server `ftp://ftp.cstug.cz/tex/CTAN/`

Během přípravy tohoto textu se autor anglické verze Tobias Oetiker dotazoval na `comp.text.tex`, a dostal mnohé odpovědi. Následující osoby pomohly s textem, jeho korekcemi a radami přispěli ke zkvalitnění tohoto textu. Autor anglické verze jim mnohokráte děkuje za cenné rady. Autor anglického textu se činí odpovědným za všechny chyby anglické verze a tvrdí, že veškerá slova, jež jsou zapsána správně, jsou jistě od lidí, kteří radili.

> Rosemary Bailey, David Carlisle, Chris McCormack, David Dureisseix, Elliot, Robin Fairbairns, Alexandre Guimond, Cyril Goutte, Neil Hammond, Rasmus Borup Hansen, Martien Hulsen, Eric Jacoboni, Alan Jeffrey, Byron Jones, David Jones, Andrzej Kawalec, Christian Kern, Jörg Knappen, Maik Lehradt, Claus Malten, Hubert Partl, John Refling, Mike Ressler, Brian Ripley, Young U. Ryu, Chris Rowley, Craig Schlenter, a Josef Tkadlec.

Při překladu do češtiny bylo neustále přihlíženo k příručce *LaTeX Stručný popis* Pavla Sýkory [10]. Tato příručka popisující starý LaTeX 2.09 byla po dlouhou dobu jedinou volně šiřitelnou dokumentací k LaTeXu v češtině; je dosud šířena spolu s instalací CSTeXu.
Autor české verze by chtěl touto cestou poděkovat P.Sýkorovi za jeho užitečnou příručku. Nemalý dík patří korektorům T. Koubovi, T. Davídkovi, A. Štědrému a M. Kočerové. Další dík patří rozhodně CSTUGu, který umožnil užití krás TeXu v našem rodném jazyce. Především je třeba poděkovat propagátorům a tvůrcům československého TeXu P. Olšákovi, J. Wágnerovi, L. Lhotkovi, J. Chlebíkové a P. Sojkovi. Zvláštní dík patří Petru Olšákovi za knihu *TeXbook naruby* [13], která je překrásnou exkurzí do tajů vnitřností TeXu.

Předmluva

LaTeX[1] je typografickým systémem, který je určen k sazbě vědeckých a matematických dokumentů vysoké typografické kvality. Systém je rovněž vhodný pro tvorbu všech možných druhů jiných dokumentů, od jednoduchých dopisů po složité knihy. Systém LaTeX je postaven na typografickém formátovacím programu TeX[2] Donalda E. Knutha.

Tento stručný úvod popisuje LaTeX 2_ε a měl by čtenáři dostačovat pro většinu aplikací LaTeXu. Úplný popis systému LaTeX lze nalézt v knihách [1, 3].

LaTeX lze užívat na širokém spektru platforem od IBM PC výše. Na většině universitních počítačových sítích, většinou na platformě operačního systému UNIX, bývá LaTeX již předinstalován. Informace o tom, jakým způsobem užívat lokální instalaci LaTeXu, by měli být v *Local Guide* [5]. Máte-li problémy začít, zeptejte se toho, kdo vám doporučil tuto brožurku, případně vašeho systémového administrátora. Mnohé otázky v češtině vám velice rádi zodpoví lidé v elektronické diskusní skupině `CsTeX@cs.felk.cvut.cz`. Účelem tohoto dokumentu *není* vysvětlovat proceduru instalace a nastavení systému. Tento úvod si však klade za cíl naučit čtenáře psaní dokumentů pomocí systému LaTeX.

Příručka je rozdělena do čtyř kapitol:

Kapitola 1 popisuje základní strukturu dokumentu systému LaTeX 2_ε. Čtenář je stručně poučen o historii LaTeXu. Po přečtení této kapitoly by měl mít čtenář hrubou představu o tom co je to LaTeX. Tato představa bude pouze rámcová a bude rozšiřována a doplňována v dalších kapitolách.

Kapitola 2 popisuje detaily sazby dokumentů. Naučí čtenáře práci se základními příkazy a prostředími LaTeXu. Po přečtení této kapitoly by měl být čtenář schopen sestavit svůj první dokument v LaTeXu.

Kapitola 3 se zabývá sazbou matematických rovnic pomocí LaTeXu. Mnohé příklady napomohou čtenáři porozumět jedné z nejmocnějších stránek LaTeXu, sazbě matematických rovnic. Na konci této kapitoly jsou uvedeny tabulky se seznamem všech matematických symbolů užívaných v LaTeXu.

Kapitola 4 rozšíří znalost některých méně užívaných, přesto užitečných, příkazů LaTeXu. Mezi jinými se čtenář naučí vkládat do dokumentu grafiku ve formátu EPS (Encapsulated PostScript) a vytvářet rejstřík.

Považujeme za důležité číst kapitoly v uvedeném pořadí. Kniha není zas tak rozsáhlá, aby se nedala přečíst celá. Zvláštní pozornosti doporučujeme řešené příklady, jelikož velká část informace je obsažena právě v nich.

Pro ty čtenáře, kteří hledají další materiály o, a k LaTeXu doporučujeme hledat v některém z ftp archivů `CTAN`. V U.S.A. je archiv na `ctan.tug.org`, v Německu na `ftp.dante.de`, ve Spojeném Království je na `ftp.tex.ac.uk` a v České republice na `ftp.cstug.cz`.

Veškeré nápady, rady, rozšíření a připomínky k tomuto textu jsou netrpělivě očekávány na adresách autorů i překladatele. Specielně autory zajímá zpětná vazba od nováčků LaTeXu, zvláště co se jim zdá býti vysvětleno málo jasně a které části by bylo dobré rozšířit.

Tobias Oetiker <oetiker@ee.ethz.ch>
Department of Electrical Engineering,
Swiss Federal Institute of Technology

Michal Kočer <kocer@ipex.cz>

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Co je dobré vědět

V první části této kapitoly je čtenáři předložen stručný přehled filosofie a historie systému LaTeX 2_ε. Druhá část kapitoly se zaměřuje na základní strukturu LaTeXovského dokumentu. Po přečtení této kapitoly by měl čtenář získat přehled o tom jak LaTeX funguje. Při dalším čtení to umožní čtenáři napojovat všechny nově získané informace do širšího kontextu.

1.1 The Name of the Game

1.1.1 TeX

TeX je počítačový program vytvořený profesorem Donaldem E. Knuthem [2]. Je určen pro sazbu textu a matematických rovnic, při zachování vysoké typografické úrovně výsledného dokumentu.

TeX vyslovujeme „tech“; „X“ pochází z řeckého χ, proto jej vyslovujeme jako „ch“. Logo TeX přepisujeme do `ASCII` jako `TeX`.

1.1.2 LaTeX

LaTeX je balík maker, který umožňuje autorům sázet a tisknout jejich díla v nejvyšší možné typografické kvalitě, přičemž autor používá profesionály předdefinovaných vzhledů dokumentů. LaTeX byl původně napsán Leslie Lamportem [1]. LaTeX užívá programu TeX jako sázecího stroje.

V současné době je LaTeX rozšiřován týmem LaTeX3, vedeným Frankem Mittelbachem. Tento tým se znaží sjednotit všechny rozšiřující verze LaTeXu, které postupně vznikaly od vzniku LaTeXu 2.09. Aby byla nějakým způsobem odlišena stará verze od nové, tak byla označena LaTeX 2_ε. Tento dokument se zaobírá verzí LaTeX 2_ε.

Slovo LaTeX se vyslovuje „lej-tech“ nebo „la-tech.“ Při odkazech na LaTeX v prostředí `ASCII` se LaTeX přepisuje jako `LaTeX`. LaTeX 2_ε se vyslovuje „lej-tech tů í“ a zapisuje se jako `LaTeX2e`.

1.2 Základy

1.2.1 Autor, typograf a sazeč

Pro publikaci předává autor nakladateli obvykle rukopis psaný na stroji. Typograf nakladatelství pak rozhodne o úpravě písemnosti (délka řádku, druh písma, odstupy před a za kapitolou atd.) a napíše sazeči k tomu nezbytné příkazy a sazeč podle těchto příkazů tiskovinu vysází.

V našem případě přebírá roli typografa LaTeX a TeX přebírá úlohu sazeče.

Typograf-člověk pozná úmysly autora (např. nadpisy kapitol, citace, příklady, vzorce, ...) z obsahu rukopisu hlavně na základě svých odborných znalostí. LaTeX naproti tomu je „jen“ program, proto zde potřebuje doplňkové informace autora, které udávají logickou strukturu textu. Tyto informace se zadávají ve formě tzv. „příkazů“ v textu.

V protikladu k tomu stojí příprava písemností pomocí programů pro zpracování textu jako je např. *Word for Windows*, *WordPerfect* nebo *Text602*. V tomto případě autor určuje úpravu textu interaktivně. Přitom vidí na obrazovce (údajně) přesně to, co bude vytištěno na stránce. Takové systémy, které podporují optický návrh, se nazývají WYSIWIG[1].

Při práci s LaTeXem autor při psaní vstupního souboru obvykle nevidí jak text bude po formátování vypadat. Výsledek však lze po zpracování LaTeXem prohlížet na obrazovce grafického terminálu případně vytisknout. A podle tohoto výstupu může svůj vstupní soubor patřičně upravit a pokračovat v práci.

1.2.2 Úprava textu

Typografický návrh je ruční práce, které se musí člověk naučit. Necvičení autoři dělají často vážné chyby formátování. Mnoho laiků se nesprávně domnívá, že tento návrh je především otázka estetiky — když dokument vypadá dobře z umělecké stránky, je dobře navržen. Poněvadž ale dokumenty nebudou pověšeny v muzeu, nýbrž je lidé budou číst, je snadná čitelnost a dobrá srozumitelnost důležitější než umělecký vzhled.

Například bychom měli zvolit takovou výška písma a číslování nadpisů, aby byla zřetelná struktura kapitol a podkapitol. Délky řádků bychom měli zvolit takové, aby nedocházelo k namáhavým pohybům očí čtenáře, nikoliv takové, aby byl papír co nejkrásněji zaplněn.

Pomocí interaktivních WYSIWYG systémů autoři obvykle snadno tvoří z estetického hlediska hezké dokumenty ale často velice špatně strukturované. LaTeX takovým chybám formátování zabraňuje, neboť je v něm autor nucen zadat logickou strukturu textu a pak pro něj použít nejvhodnější formát.

[1] z angl. *what you see is (all) what you get*

1.2.3 Výhody a nevýhody

Častým tématem vášnivých diskusí milovníků WYSIWYG systému a milovníků LaTeXu jsou výhody případně nevýhody LaTeXu oproti normálním textovým procesorům. Nejlepší věc, kterou lze udělat, když taková diskuse začíná, je držet se zpět, jelikož se taková diskuse velice často vymkne člověku z rukou. Avšak jsou chvíle, kdy se nedá nikam utéct...

Oproti ostatním textovým procesorům se LaTeX vyznačuje následujícími výhodami:

- Je k dispozici několik profesionálně vytvořených formátů, se kterými dokumenty vypadají „jako profesionálně vytištěné".
- Zvlášť dobře je podporována sazba matematických vzorců.
- Uživatel musí zadávat jen několik lehce srozumitelných příkazů, které se týkají logické struktury dokumentu, a (téměř) nepotřebuje zabývat se technickými detaily tisku.
- Bez velké námahy mohou být vytvořeny také složité struktury jako poznámky pod čarou, seznamy literatury, obsahy, tabulky atd. stejně jako odkazy na stránku, čísla kapitol tabulek, obrázků, rovnic atd.
- Pro mnoho typografických funkcí, které nejsou přímo podporovovány základním LaTeXem exitují snadno připojitelné nadstavby, tzv. balíky maker. Například lze používat balík umožňující práci z grafickým formátem ve formě PostScriptu či balík pro sazbu bibliografie splňující přesně daný standard. Většina podobných balíků je popsána v *The LaTeX Companion* [3].
- TeX, základní program pro práci LaTeXu, je přenositelný na všechny užívané operační systémy[2], proto jej lze používat na libovolném hardware, a navíc TeX i LaTeX jsou i přes své vysoké kvality zdarma.

LaTeX má i své nevýhody:

- Dokonalejší algoritmy pro formátování kladou větší nároky na čas zpracování a paměť než u jednodušších programů pro zpracování textu. Kompletní instalace zabírá poměrně mnoho diskového prostoru. Avšak vzhledem k tomu, že v současné době se objevují textové procesory Word for Windows 6.0 užívající daleko více diskové kapacity než běžný LaTeX, není tento zápor nutné brát v úvahu. Pokud se týče využití procesoru (CPU) LaTeX poráží každý WYSIWYG systém, jelikož potřebuje procesor pouze v okamžiku zpracovávání zdrojového textu,

[2] TeX lze provozovat dokonce i na systémech takového typu jeko je MS-DOS či MS-Windows

zatímco WYSIWYG spotřebovává čas procesoru po celou dobu své činnosti.

- Výstup textu je možný pouze na grafických zařízeních (laserové, inkoustové nebo jehličkové tiskárny, grafické obrazovky), nikoliv na levných znakově orientovaných rychlotiskárnách.
- Ačkoli lze v předem připravených dokumentních stylech snadno přizpůsobit jednotlivé parametry, zásadní změny předem stanovených formátů vyžadují větší námahu (návrh nového dokumentního stylu).[3]

1.3 Vstupní soubory LaTeXu

Vstupním souborem LaTeXu je čistý textový `ASCII` soubor, vytvořený libovolným textovým editorem. Tento soubor obsahuje kromě vlastního textu, který má být vytištěn, i „příkazy“, které napoví LaTeXu, jak má text vysázet.

1.3.1 Mezery

„Neviditelné“ znaky jako mezera (blank), tabulátor či konec řádku (carriage return) jsou zpracovány LaTeXem jako mezera. *Několik* po sobě následujících neviditelných znaků, mezer, je zpracováno jako *jedna* mezera. Neviditelné znaky na počátku řádky jsou LaTeXem ignorovány.

Prázdný řádek mezi textovými řádky označuje konec odstavce. *Několik* prázdných řádků je zpracováno jako *jeden* prázdný řádek.

Následuje příklad. Napravo je text vstupního textového souboru a nalevo je LaTeXem formátovaný výstup.

Je naprosto jedno, zda vložíme jednu či několik mezer mezi slovy. Prazdný řádek uvozuje nový odstavec.	`Je      naprosto     jedno,      zda` `vložíme jednu  či  několik   mezer` `mezi                         slovy.` `Prazdný       řádek        uvozuje` `  nový                  odstavec.`

1.3.2 Speciální znaky

Následující symboly jsou rezervované znaky, které mají v LaTeXu zvláštní význam nebo je nelze použít ve všech druzích písma. Napíšete-li některý z nich přímo do vstupního souboru, obvykle se na výstupu neobjeví, ale pravděpodobně donutí LaTeX dělat něco co jste vůbec nechtěli.

```
$ & % # _ { } ~ ^ \
```

[3]Říká se, že právě tento problém má vyřešit vznikající systém LaTeX3

Následující znaky mohou být vytištěny zadáním dvojice znaků „obrácené lomítko“[4] (\) a požadovaného znaku:

$ & % # _ { }	`\$ \& \% \# \_ \{ \}`

Zbývající symboly a ještě mnoho dalších znaků lze vytisknout speciálními příkazy jako akcenty (diakritická znaménka) nebo v matematických vzorcích[5].

1.3.3 Příkazy LaTeXu

V příkazech LaTeXu se rozlišují malá a velká písmena (case sensitivity) a většinou mají následující formát:

- Začínají znakem zpětné lomítko \ následovaným jménem příkazu složeným pouze z písmen. Jméno příkazu pak končí mezerou nebo prvním znakem, jež není písmenem, například číslicí či jiným nealfanumerickým znakem.
- Zkládají se ze znaku zpětné lomítko následovaným právě jedním zvláštním znakem

LaTeX ignoruje veškeré neviditelné znaky, např. mezery, za příkazy. Chcete-li získat za příkazem mezeru je třeba příkaz ukončit buď znaky `{}` následovanými mezerou, nebo specielním příkazem pro vložení mezery následujícím za jménem příkazu.

Donald Knuth píše, že lidé pracující s programem TeX se dělí na TeXniky a TeXperty.

Dnes je 7. března 1997.

Nebo: Dnes je 7. března 1997.

Špatně: Dne 7. března 1997prší.

Správně: Dne 7. března 1997 sněží.

Nebo: Dne 7. března 1997 neprší.

```
Donald Knuth píše, že lidé
pracující s programem \TeX{}
se dělí na
\TeX{}niky a \TeX perty.\\

Dnes je \today.

Nebo: Dnes je \today .

Špatně: Dne \today prší.

Správně: Dne \today{} sněží.

Nebo: Dne \today\ neprší.
```

Některé příkazy vyžadují ke své činnosti ještě parametr, jež se uzavírají do složených závorek `{ }` těsně za jménem příkazu. Některé příkazy

[4]angl. *backslash*

[5]V LaTeXu existuje také prostředí `verbatim`, ve kterém se všechny znaky včetně vícenásobných mezer opisují do výstupního textu.

umožňují navíc zadávat nepovinné parametry, které se zadávají za jménem příkazu do hranatých závorek `[ ]`. Následující příklad ukazuje užití několika příkazů LaTeXu. Význam těchto příkazů bude vysvětlen později.

Můžete se o mě *opřít*!

```
Můžete se o mě \textit{opřít}!
```

Prosím, začněte novou řádku tady a hned!
Děkuji!

```
Prosím, začněte novou řádku
tady a hned!\linebreak[4]
Děkuji!
```

1.3.4 Komentáře ve vstupním souboru

Narazí-li LaTeX ve vstupním souboru na znak `%` vše co stojí za tímto znakem je až do konce řádku ignorováno. To se autorům hodí pro vkládání komentářů, jež nemají být součástí výstupu, do vstupních souborů.

To je příklad.

```
To je % hloupý
% Lépe: poučný <----
příklad.
```

1.4 Struktura vstupního souboru

Když LaTeX 2_ε zpracovává vstupní soubor očekává, že tento soubor bude mít pevně danou structuru. Proto každý vstupní soubor musí začínat příkazem

```
\documentclass{...}
```

Tímto příkazem autor specifikuje, jakého druhu bude vytvářený dokument. Za tímto příkazem mohou následovat příkazy, které mají vliv na vzhled celého dokumentu, případně lze načíst balík maker příkazem:

```
\usepackage{...}
```

Balíky maker lze rozšířit chování LaTeXu o nové funkce, vlastnosti a příkazy.

Je-li vše potřebné nastaveno[6] lze příkazem

```
\begin{document}
```

označit začátek vlastního těla dokumentu.

Tělo dokumentu obsahuje vlastní text dokumentu promýchaný s příkazy LaTeXu. Tělo dokumentu se ukončí příkazem

```
\end{document}
```

[6] Oblast mezi `\documentclass` a `\begin{document}` je označována *preambule*

Tím je LaTeXu řečeno, že je to vše co jsme po něm chtěli. Vše co následuje za tímto příkazem je proto LaTeXem ignorováno.

Obrázek 1.1 ukazuje minimální vstupní soubor pro LaTeX 2$_\varepsilon$. Poněkud složitější vstupní soubor je uveden na obrázku 1.2.

1.5 Struktura dokumentu

1.5.1 Třídy dokumentů (Document Classes)

První informací, kterou LaTeX potřebuje vědět při zpracovávání vstupního souboru je informace o typu dokumentu, který autor hodlá vytvářet. Tuto informaci autor předá LaTeXu právě příkazem `\documentclass`.

`\documentclass[`*options*`]{`*class*`}`

Kde *class* vyznačuje typ (třídu) vytvářeného dokumentu. V tabulce 1.1 je uveden přehled tříd dokumentu. Distribuce systému LaTeX 2$_\varepsilon$ zavádí další

```
\documentclass{article}
\begin{document}
Co je malé, to je hezké.
\end{document}
```

Obrázek 1.1: Minimální soubor LaTeXu

```
\documentclass[a4paper,11pt]{article}
\usepackage{czech}
\usepackage{latexsym}
\author{A.~Utor}
\title{Krátce}
\frenchspacing
\begin{document}
\maketitle
\tableofcontents
\section{Úvod}
Tady začíná můj skvělý článeček\ldots{}
\section{Závěr}
\ldots{} a tady je úplný konec.
\end{document}
```

Obrázek 1.2: Příklad odborného článku

rozšíření tříd dokumentu o dokumenty dopisů (letter) a blán (slides). Nepovinný parametr *option* blíže určuje vlastnosti třídy dokumentu. Lze dokonce užít několika nepovinných parametrů najendou s tím, že je oddělíme čárkou. V tabulce 1.2 jsou uvedeny nejužívanější volitelné parametry pro standardní třídy dokumentů.

Příklad: Vstupní soubor pro LaTeX může začít, například, takovouto řádkou:

```
\documentclass[11pt,twoside,a4paper]{article}
```

Tím se LaTeX dozví, že bude sázet *článek* (article) o základní velikosti písma *jedenáct bodů* (11pt) a bude formátovat výstup tak, aby byl vhodný pro *oboustranný* (double sided) tisk na stránku o velikosti *a4*.

1.5.2 Balíky maker

Při psaní dokumentu se někdy stane, že základní prostředky LaTeXu neumožní vyřešit nějaký problém. Je-li třeba do dokumentu vložit grafiku, barevný text nebo zdrojový text nějakého programu je potřeba nějakým způsobem rozšířit schopnosti LaTeXu. Taková rozšíření se nazývají balíky maker (packages). Balík maker se aktivuje užitím příkazu

`\usepackage[`*options*`]{`*package*`}`

Kde *package* je jméno balíku maker a *options* je seznam klíčových slov, které zjemňují, upřesňují či spouštějí některé specielní rysy balíku maker. Některé balíky maker jsou nedílnou součástí každé distribuce systému LaTeX 2_ε (viz tabulku 1.3). Mnohé balíky jsou však distribuovány samostatně. Více informací o naistalovaných balících maker se lze dočíst v lokálním průvodci instalací *Local Guide* [5]. Hlavním zdrojem informací o LaTeXu je *The LaTeX*

Tabulka 1.1: Třídy dokumentů

`article` pro odborné články, prezentace, krátké zprávy, dokumetaci k pagramům, pozvánky, ...

`report` pro delší zprávy obsahující několik kapitol, menší knihy, diplomové práce, ...

`book` pro skutečné knihy

`slide` pro blány (slides). Tato třída užívá velkého bezpatkového písma, jež je pro zpětnou projekci nejvhodnější.

Tabulka 1.2: Nepovinné paramtry tříd dokumentů

`10pt`, `11pt`, `12pt` Nastavuje velikost hlavního písma dokumentu. Není-li tato volba explicitně uvedena použije se pro základní písmo velikost `10pt`.

`a4paper`, `letterpaper`, ... Nastavuje velikost stránky, na níž je provedena sazba výsledného dokumentu. Základní velikost je `letterpaper`. Mimo to lze užít `a5paper`, `b5paper`, `executivepaper`, a `legalpaper`.

`fleqn` Matematické rovnice se budou zarovnávat vlevo namísto do středu.

`leqno` Čísla rovnic se nebudou sázet vpravo, ale vlevo na stránku.

`titlepage`, `notitlepage` Určuje zda se bude sázet samostatná titulní strana či nikoli. Třída `article` neužívá samostatnou titulní stranu zatímco třídy `report` a `book` tak činí.

`twocolumn` LaTeX bude sázet dokument do dvou sloupců. .

`twoside, oneside` Určuje zda má být generován jedno- či oboustranný výstup. Třídy `article` a `report` normálně užívají jednostranné a třída `book` oboustranné sazby. Jednostranná sazba od oboustranné se liší rozdílnou velikostí okraje pravé a levé strany pro vazbu případně umístěním čísla stránky.

`openright, openany` Specifikuje zda má nová kapitola začít vždy na nové pravé stránce či není-li to nutné. Tato volba nespolupracuje s třídou `article`, jelikož tato třída nerozpoznává pojem kapitola (chapter). Třída `report` normálně začíná novou kapitolu na další volné stránce (je jedno zda pravé či levé) a třída `book` začíná sázet novou kapitolu na nejbližší pravé stránce.

Companion [3]. Tato kniha obsahuje spolu s popisem více než stovky balíků maker i popis toho jakým způsobem tvořit vlastní rozšíření LaTeX 2_ε.

Tabulka 1.3: Balíky distribuované spolu s LaTeXem

`doc` Umožňuje přehledné dokumentování LaTeXovských maker, programů.
Balík je popsán v `doc.dtx`[a] a v *The LaTeX Companion* [3].

`exscale` Umožňuje zvětšovatelné verze matematických fontů.
Popsán v `ltexscale.dtx`.

`fontenc` Specifikuje jaké kódování fontů má LaTeX užít.
Popsán v `ltoutenc.dtx`.

`ifthen` Umožňuje příkazy typu ve formě
'if... then do... otherwise do....'
Popsán v `ifthen.dtx` a *The LaTeX Companion* [3].

`latexsym` Ke spřístupnění fontů symbolů LaTeXu. Popsán v `latexsym.dtx` a v *The LaTeX Companion* [3].

`makeidx` Umožňuje pomocí rozšířených příkazů jednoduše vytvářet restříky. Popsán v oddíle 4.5 a v *The LaTeX Companion* [3].

`syntonly` Umožňuje zpracovat dokument bez vysázení.
Popsán v `syntonly.dtx` a v *The LaTeX Companion* [3].
Tento styl se hodí pro rychlou kontrolu chyb.

`inputenc` Umožňuje specifikovat kódování vstupu jako ASCII, ISO Latin-1, ISO Latin-2, 437/850 IBM code pages, Apple Macintosh, Next, ANSI-Windows či uživatelem definované.
Popsán v `inputenc.dtx`.

[a] Tento soubor my měl být instalován spolu s vaším systémem. Dokumentaci ve formě souboru `dvi` získáte jednoduše „přeložením“ LaTeXem, obvykle příkazem systému `latex doc.dtx`. To samé platí i pro další balíky v této tabulce.

1.5.3 Styly stránek

LaTeX nabízí tři předdefinované kombinace záhlaví/paty stránek — tzv. stránkové styly. Parametr *style* příkazu

`\pagestyle{`*style*`}`

definuje, který ze stránkových stylů se užije. Tabulka 1.4 dává přehled o předdefinovaných stylech stránek.

Tabulka 1.4: Předdefinované styly LaTeXu

`plain` tiskne čísla stránek na spodním okraji stránky ve středu paty stránky. Toto je základní stánkový styl.

`headings` tiskne jméno aktuální kapitoly a číslo stránky v záhlaví každé stránky a pata stránky zůstává prázdná. (To je style stránky použitý v tomto dokumentu)

`empty` nastavuje prázdné záhlaví i patu stánky.

Změnu stylu aktuální stránky lze provést příkazem

`\thispagestyle{`*style*`}`

V *The LaTeX Companion* [3] je uveden popis toho jak lze tvořit vlastní vzhled stránky.

1.6 Rozsáhlé projekty

Pracujete-li na rozsáhlém dokumentu, je dobré rozdělit vstupní soubor na několik částí. LaTeX poskytuje dva příkazy, které autorům pomáhají v tomto případě.

`\include{`*filename*`}`

tento příkaz se užije v těle dokumentu, v místě kam chceme vložit obsah jiného souboru. Je třeba podotknout, že LaTeX začne novou stránku před tím než začne spracovávat vstupní soubor *filename*.

Druhý příkaz se užívá v preambuli dokumentu. Umožňuje autorovi nařídit LaTeXu vložit pouze některé ze souborů vkládaných příkazem `\include`.

`\includeonly{`*filename*`,`*filename*`,...}`

Poté co se provede v preambuli dokumentu tento příkaz jsou dále provedeny příkazy `\include` pouze pro jména souborů (*filename*), která jsou uvedena v argumentu příkazu `\includeonly`. Je třeba si dát pozor na to, že mezi čárkou a jménem souboru nesmí být mezera.

Po provedení příkazu `\include` se vkládaný text začne sázet na nové stránce. To je užitečné, když se užije příkaz `\includeonly`, jelikož stránkové zlomy se nepřesouvají ani, když se některé vkládané soubory přeskočí. Avšak někdy tato vlastnost není příjemná. V takovém případě lze užít příkazu

`\input{`*filename*`}`

Tento příkaz jednoduše vloží soubor *filename* do hlavního vstupního souboru.

Kapitola 2

Sázení textu

Předchozí kapitola seznámila čtenáře se základními pojmy tvorby dokumentu v LaTeXu. Tato kapitola slouží k prohloubení těchto znalostí a v něterých směrech je rozvíjí tak, aby byl čtenář schopen sázet dokumenty použitelné k běžné práci.

2.1 Řádkový a stránkový zlom

2.1.1 Zarovnání odstavců

Většina knih je sázena tak, že všechny řádky mají stejnou délku. LaTeX sám vkládá potřebné řádkové zlomy a mezery mezi slova a optimalizuje naráz vzhled celého odstavce. V případech, kdy nemůže vhodně vložit řádkové zlomy mezi slova, začne automaticky podle předepsaných pravidel slova dělit. To jak bude nakonec vysázený odstavec vypadat záleží i na třídě sázeného dokumentu, tj. na parametru příkazu `\documentclass`. V běžných případech je první řádek odstavce odsazen a mezi jednotlivými odstavci není vynechané místo; viz též oddíl 4.2.2.

Ve výjimečných případech je však třeba LaTeXu řádkový zlom nařídit příkazem

`\\` či `\newline`

nařídí vložení řádkového zlomu bez toho, aby začal nový odstavec.

`\\*`

navíc zakáže stránkový zlom po tomto řádkovém zlomu.

`\newpage`

nařídí v daném místě dokumentu stránkový zlom.

`\linebreak[n]`, `\nolinebreak[n]`, `\pagebreak[n]` and `\nopagebreak[n]`

dělají přesně to co říkají jejich anglická jména. Dávají autorům do ruky možnost ovlivňovat svoji činnost pomocí nepovinného parametru n. Tento parametr může nabývat hodnot nula až čtyři. Nastaví-li se n na hodnotu nižší než 4 nemusí LaTeX žádanou akci provést, kdyby výsledek sazby nebyl hezký.

LaTeX vždy zkouší provádět řádkové zlomy v co nejvhodnějším místě. Nelze-li nalézt vhodné místo pro zlom řádky způsobem, který by splňoval poměrně vysoké nároky na řádkové zlomy, ponechá se jeden řádek přečnívat vpravo z odstavce a LaTeX vypíše chybové hlášení („overfull hbox"). To se stává zejména tehdy, když LaTeX nemůže nalézt vhodné místo pro rozdělení některého slova. Zadáním příkazu `\sloppy` lze LaTeXu nařídit, aby snížil nároky na kvalitu sazby. LaTeX poté řeší tento problém tak, že vkládá větší mezery mezi slovy i v případě, že konečný výstup není optimální. V případě, že TeX není spokojen se sazbou, uvědomí uživatele upozorněním („underfull hbox"). V mnoha případech však výsledek nevypadá úplně nejhůře. Příkaz `\fussy` pracuje zcela opačně, užívá se v případě, kdy chceme, aby si LaTeX stěžoval úplně na vše.

2.1.2 Dělění slov

V případě potřeby LaTeX automaticky dělí slova. Pokud jednotlivé výsledky automatického dělení slov nejsou správné, můžete tyto „výjimky" napravit níže uvedenými příkazy[1].

Příkaz

`\hyphenation{word list}`

vymezuje slova, která lze dělit pouze v místech označených „`-`". Tento příkaz je třeba užít v preambuli vstupního souboru a může obsahovat slova složená pouze z písmen, přičemž se ignoruje jejich velikost. V níže uvedeném příkladu je dovoleno dělení slova „rozdělení" stejným způsobem jako slova „Rozdělení" a zakázáno dělení nejen slova „FORTRAN", ale i „Fortran" či „fortran". V argumentu příkazu nelze užívat speciální znaky a symboly.

Příklad:

```
\hyphenation{FORTRAN Roz-dě-le-ní}
```

Příkaz `\-` umožňuje označit místo pro rozdělení slova. Takto označené místo se pro dané slovo stane jediným možným místem pro rozdělení. Tento způsob je vhodný především pro slova, která obsahují speciální znaky, jelikož LaTeX taková slova automaticky nedělí.

[1] Obzvláště to může být nutné při použití cizojazyčných slov nebo např. u složených slov v němčině.

Jedno z nejdelších slov je: nejneobhospodařovatelnějšími

```
Jedno z~nejdelších slov je: nej\-%
ne\-ob\-hos\-po\-da\-řo\-va\-%
tel\-něj\-ší\-mi
```

Několik slov lze udržet pospolu na stejné lince příkazem

`\mbox{`*text*`}`

Příkaz zajistí, že jeho argumenty budou za všech okolností považovány LaTeXem za jedno nedělitelné slovo.

Mé telefonní číslo se brzy změní na +420 38 570 91

```
Mé telefonní číslo se brzy změní na
\mbox{+420 38 570 91}
```

Parametr *filename* určuje jméno souboru.

```
Parametr
\mbox{\emph{filename}} určuje
jméno souboru.
```

2.2 Speciální znaky a symboly

2.2.1 Uvozovky

Píšete-li text na psacím stroji, můžete jako uvozovky použít pouze znak „uvozovky nahoře" (`"`). V knihtisku se však otevírací a uzavírací uvozovky sázejí různými znaky. Navíc jsou otevírací i uzavírací uvozovky v anglickém textu odlišné od příslušných uvozovek v textu českém, slovenském, německém apod. Ve vstupním souboru s anglickým textem se otevírací uvozovky zadávají jako dva znaky „obrácený apostrof" (`` ` ``), uzavírací jako dva znaky „apostrof" (`'`). Zapamatujte si, že uvozovky v textu se *nikdy* nezadávají pomocí klávesy `"`. Při úpravách textů původně určených pro jiný sázecí systém nebo textový procesor nezapomeňte takto zadané uvozovky opravit. Špatně zadané uvozovky nevypadají v hotovém dokumentu vůbec dobře.

“Please press the ‘x’ key.”

```
``Please press the `x' key.''
```

Uvozovky v češtině, němčině apod. vypadají jinak a mohou se zadávat několika způsoby (viz kap. 2.4.3).

2.2.2 Pomlčky a podobné znaky

V typografii se užívají pomlčky různých délek. Nejkratší je spojovník (`-`), pak následuje krátká půlčtverčíková pomlčka (`--`), pak dlouhá čtverčíková pomlčka (`---`). Odlišné je i matematické znaménko „mínus" (`$-$`).

Chceme-li, aby hotový dokument vypadal esteticky, je nezbytné pomlčky používat správně. Krátkou pomlčku používáme pro znázornění intervalu.

Od ostatního textu ji neoddělujeme mezerou[2]. Dlouhá pomlčka naznačuje přestávku v řeči nebo od sebe výrazně odděluje části textu; významově je rovna čárce[3]. V českých a německých textech se sází s mezerami z obou stran, v anglických bez mezer. Při řádkovém zlomu zůstává na konci řádku, nový řádek jí nesmí začínat.

půjde-li to	`půjde-li to\\`
10–16 hodin	`10--16~hodin\\`
yes—or no?	`yes---or no? \\`
0, 1 či -1	`$0$, $1$ či $-1$`

2.2.3 Výpustky, elipsy (. . .)

Na rozdíl od strojopisného textu, kde každá tečka nebo čárka zabírá stejnou šířku jako ostatní znaky, při knihtisku se tečky a čárky sázejí užší. Pro tři tečky ve smyslu „a tak dále“, tzv. „elipsy“, se zde používá příkazů

`\ldots`

Takto ne ... nýbrž takto:	`Takto ne ... nýbrž takto: \\`
Praha, Plzeň, . . .	`Praha, Plzeň,~\ldots`

Před výpustku se sází normání nebo zúžená mezera.

2.2.4 Slitky a podřezávání (kerning)

V knižním tisku je zvykem sázet některé kombinace písmen jako jeden znak — tzv. *slitky* neboli *ligatury.*

ﬀ ﬁ ﬂ ﬃ . . . namísto ff fi fl ffi . . .

Podřezávání (kerning) je přiblížení některých dvojic znaků k sobě (přesněji řečeno přes sebe), aby text opticky vypadal lépe.

AV Te . . . místo A V T e . . .

LaTeX (resp. TeX) vytváří slitky i podřezávání automaticky. Slitky se však nepoužívají ve složených slovech (nebo ve slovech s předponou či příponou), jejichž části se stýkají kombinací písmen ﬀ, ﬂ apod. Toto se týká pouze několika velmi málo anglických slov a nemnoha slov německých. V takovém případě lze tvorbě slitků zamezit například vložením `\mbox{}` nebo `{}` mezi písmena, která by mohla ligaturu utvořit.

[2] Někdy kolem půlčtverečníkové pomlčky sází zúžená mezera (`\,`)

[3] Tato pomlčka přežívá z viktoriánské éry a je postupně nahrazována půlčtverečníkovou pomlčkou oddělenou od ostatního textu mezerou

Nikoli shelfful
nebo Auflage (Au-fl-age)
ale shelfful
nebo Auflage (Auf-lage)

```
Nikoli shelfful \\
nebo Auflage (Au-fl-age)\\
ale  shelf\mbox{}ful \\
nebo Auf{l}age (Auf-lage)
```

2.2.5 Akcenty a speciální znaky

LaTeX umožňuje použít akcenty (diakritická znaménka) a zvláštní znaky mnohých jazyků (viz tab. 2.1). Akcenty jsou zde demonstrovány na příkladě písmena o, ovšem principiálně není vyloučeno jejich použití s kterýmkoli jiným písmenem. Pokud má být akcent vysázen nad písmenem i nebo j, musí být použito písmeno „i bez tečky“ nebo „j bez tečky“. Tyto znaky se vysázejí pomocí příkazu `\i` nebo `\j`.

Hôtel, naïve, élève,
smørrebrød, ¡Señorita!,
Schönbrunner Schloß Straße

```
H\^otel, na\"\i ve, \'el\`eve,\\
sm\o rrebr\o d, !`Se\~norita!,\\
Sch\"onbrunner Schlo\ss{}
Stra\ss e
```

Tabulka 2.1: Akcenty a zvláštní znaky

ò	\`o	ó	\'o	ô	\^o	õ	\~o
ō	\=o	ȯ	\.o	ö	\"o		
ŏ	\u o	ǒ	\v o	ő	\H o	o̧	\c o
ọ	\d o	o̲	\b o	o͡o	\t oo		
œ	\oe	Œ	\OE	æ	\ae	Æ	\AE
å	\aa	Å	\AA				
ø	\o	Ø	\O	ł	\l	Ł	\L
ı	\i	ȷ	\j	¡	!`	¿	?`

2.3 Mezislovní mezery

Aby bylo dosaženo zarovnaného pravého okraje, LaTeX zvětšuje nebo zmenšuje mezery mezi slovy. Při sazbě anglických textů navíc sází po tečkách, otaznících apod. na konci věty větší mezeru[4], což zvyšuje čitelnost textu. V českých a německých textech se sází mezery stejné. LaTeX předpokládá,

[4] Jedná se opět o historický přežitek

že tečka, která následuje za velkým písmenem, označuje zkrácení (např. křestního jména), a že všechny ostatní tečky ukončují větu.

Výjimky z těchto pravidel musíme LaTeXu sdělit následujícími příkazy: Obrácené lomítko následované mezerou označuje, že mezera ve výsledném textu v tomto místě nesmí být širší. Znak ~ (vlnka, tilda) označuje mezeru, která nesmí být širší a v tomto místě nesmí dojít k řádkovému zlomu. V českých textech bychom neměli nechávat jednohláskové předložky a spojky i, I, A, na konci řádku. Za těmito spojkami a předložkami tedy místo mezery píšeme vlnku. Pan Petr Olšák napsal program `\vlna`, který tuto činnost provádí automaticky. Další případ nepovoleného řádkového zlomu, kdy tedy musíme použít vlnku, je například mezera mezi číselnou hodnotou a jednotkou, mezera mezi titulem nebo zkratkou jména a příjmením apod. Příkaz `\@` před tečkou označuje, že touto tečkou končí věta, ačkoli je před ní velké písmeno.

Firma Čtverec a spol. Vám dodá kval. kolečka.	`Firma Čtverec a spol.\ Vám dodá kval.\ kolečka.  \\`
… 5 m široký.	`\dots\ 5~m široký. \\`
Dr. Šedý bydlí v 1. patře.	`Dr.~Šedý bydlí v~1.~patře. \\`
Potřebuji vitamín C. Ty ne?	`Potřebuji vitamín~C\@. Ty ne?`

Příkazem

```
\frenchspacing
```

LaTeXu oznámíme, aby nevkládal na konec věty dodatečnou mezeru. Toho se užívá při sazbě v neanglickém jazyce, v tomto případě není nutno používat příkazu `\@`.

Při vytváření českých textů se používá předdefinovaný formát `czech.sty`, který příkaz `\frenchspacing` aktivuje automaticky. K obnovení konvence sázení mezer anglických textů slouží příkaz

```
\nonfrenchspacing
```

2.4 Sazba českých textů

V této kapitole je popsáno zpracování českých textů systémem CSTeX (systém TeX upravený pro češtinu), který šíří Československé sdružení uživatelů TeXu (CSTUG) pro všechny nejčastěji užívané platformy. Bližší informace o CSTeXu naleznete na `http://www.cstug.cz/`.

Originální americká verze TeXu podporuje pouze dokumenty v angličtině. CSTeX podporuje standardně češtinu a angličtinu, navíc jej lze snadno upravit pro podporu dokumentů v češtině, angličtině i němčině.

Při zpracování dokumentů různými instalacemi LaTeXu bude nutné připojit použité nestandardní styly.

2.4.1 Dělení slov

Při spuštění LaTeXu je určeno, zda se bude dělit podle pravidel českého, anglického, německého případně jiného jazyka. Výjimky lze v daném dokumentu zadat pomocí příkazů uvedených v kap. 2.1.2. Při správném vytváření vícejazyčných dokumentů se pravidla pro dělení přepínají podle příslušného jazyka.

2.4.2 Písmena s akcenty

Původní verze LaTeXu obsahuje jak příkazy pro umístění akcentů nad (v některých jazycích i pod) písmena, tak i umožňuje použití písmen speciálních. Nicméně používat tyto příkazy je dosti nepohodlné. V dokumentech zpracovávaných LaTeXem obsaženým v CSTeXu lze proto při použití `\usepackage{ czech}` v preambuli vstupního souboru používat rozšířenou sadu ASCII znaků. Toto řešení umožňuje použít dělicí algoritmus a příkaz `\hyphenation` i na slova s diakritickými znaménky.

2.4.3 Uvozovky

Uvozovky v českém textu vypadají jinak než v anglickém. Pokud použijeme stylu `czech.sty`, můžeme použít pro české uvozovky příkaz

`\uv{`*text, který má být v uvozovkách*`}`

Pro anglické uvozovky platí původní příkazy — ` `` ` pro otevírací a `''` pro zavírací. Jinou možností je na začátku textu použít příkaz `\csprimeson`, po kterém příkazy ` `` ` (dva obrácené apostrofy) a `''` (dva apostrofy) vytvářejí uvozovky české. V textu se ovšem mohou vyskytnout slova s apostrofem (např. l'Hospitalovo pravidlo), který by byl v tomto případě vysázen nesprávně. Pak je možné použít příkaz `\csprimesoff`, který obnoví původní chování příkazů ` `` ` a `''` a tedy i apostrofu. Jiné možnosti vytváření a kombinací anglických a českých uvozovek jsou uvedeny v dokumentaci ke stylu `czech.sty`

„Ne," řekl, „nevím nic!"	`\uv{Ne,} řekl, \uv{nevím nic!}\\`
Jiný způsob: „uvozovky".	`Jiný způsob: \clqq uvozovky\crqq.`

2.4.4 Nadpisy a data

V originální verzi LaTeXu jsou použity pro nadpisy kapitol, obsahu, označení tabulek a obrázků a názvy měsíců v datech anglické termíny. Jejich použití by bylo v českém textu krajně nevhodné. Použitím stylu `czech.sty` se tyto názvy automaticky změní na jejich české ekvivalenty.

2.5 Nadpisy, kapitoly a jejich dělení

Pro lepší orientaci čtenáře v autorově díle, je vhodné dílo rozdělit na kapitoly, podkapitoly a oddíly. LaTeX umožňuje označit logickou strukturu díla speciálními příkazy, které jako argument mají název daného logického oddílu díla. Je pouze na autorovi, aby tyto příkazy užil ve správném pořadí.

Ve třídě `article` lze užít následujích příkazů vymezujících logickou strukturu článku:

```
\section{...}            \paragraph{...}
\subsection{...}         \subparagraph{...}
\subsubsection{...}      \appendix
```

Třídy `report` a `book` rozpoznávají navíc tyto příkazy:

```
\part{...}               \chapter{...}
```

Velikost mezer mezi jednotlivými oddíly velikost fontů jednotlivých úrovní nadpisů je nastavena automaticky LaTeXem.

Dva z těchto příkazů mají poněkud jsou poněkud zvláštní:

- Příkaz `\part` nemá vliv na číslování pořadí kapitol.
- Příkaz `\appendix` nemá žádný argument. Pouze změní označení kapitol (pro třídu `article` označení oddílů) z čísel na písmena.

Příkaz

```
\tableofcontents
```

zajistí vytištění obsahu. Nadpisy a čísla stran se dozví LaTeX z předcházejícího zpracování vstupního souboru. Proto při úpravách nebo doplnění kapitoly je nutno zpracovat dokument programem LaTeX dvakrát (někdy i třikrát), aby bylo dosaženo správného výstupu.

Všechny příkazy pro logické dělení dokumentu existují také v tzv. ohvězdičkované verzi, tzn. za jméno příkazu se přidá znak hvězdička (`*`). Jméno oddílu vytvořeného příkazem v ohvězdičkované verzi se neočísluje a neuvede v obsahu. Například příkaz `\section{Předmluva}` v ohvězdičkované verzi bude vypadat takto `\section*{Předmluva}`

Standardně jsou názvy oddílů v obsahu uvedeny přesně v té samé formě jako v textu. Někdy je však název daného oddílu příliš dlouhý na to, aby byl ve stejné formě uveden v obsahu. Proto lze položku, tak jak má být vysázena v obsahu, specifikovat jako volitelný parametr daného příkazu pro strukturalizaci.

```
\chapter[Položka do obsahu]{Zde je velmi dlouhý
                a obzvláště nudný nadpis}
```

Nadpis celého dokumentu se vysází příkazem

`\maketitle`

přičemž obsah titulní strany nebo nadpisu musí být předem zadán příkazy:

`\title{...}`, `\author{...}` a volitelně i `\date{...}`

před voláním příkazu`\maketitle`. Jako argument příkazu `\authors` lze zadat několik jmen oddělenných příkazem `\and`.

Příklad některých výše uvedených příkazů lze nalézt na obrázku 1.2 na straně 7.

Vedle již vysvětlených příkazů pro logické dělení dokumentu LaTeX 2_ε poskytuje ve třídě `book` další tři příkazy:

`\frontmatter`, `\mainmatter` and `\backmatter`

ty slouží pro hrubší rozdělení knihy. Příkazy mění nadpisy a číslování stránek způsobem u knih běžným.

2.6 Křížové odkazy

V knihách, referátech a článcích jsou velice běžné křížové odkazy na obrázky, tabulky, rovnice či jiné oddíly textu.

LaTeX nabízí pro křížové odkazy pohodlný aparát, totiž příkazy:

`\label{`*návěští*`}`, `\ref{`*návěští*`}` a `\pageref{`*návěští*`}`

Kde *návěští* je jednoznačné označení vybrané uživatelem. LaTeX zamění `\ref` číslem takového oddílu, pododdílu, obrázku, tabulky či rovnice, jež byl označen odpovídajícím příkazem `\label`, tj. příkazem `\label`, který má jako argument stejné návěští jako příslušný odkaz `\ref`. Namísto `\pageref` se vysází číslo stránky na níž se nachází odpovídající příkaz `\label`. Čísla stránek LaTeX získává z pomocného souboru vytvořeného v předchozím průběhu překladu programem TeX, proto i při použití křížových odkazů je někdy třeba překládat vstupní soubor programem TeX několikrát.

Odkaz na tento pododdíl vypadá takto: „viz též oddíl 2.6 na straně 22.“

```
Odkaz na tento pododdíl
\label{sec:tato} vypadá takto:
\uv{viz též oddíl~\ref{sec:tato} na
straně~\pageref{sec:tato}.}
```

2.7 Poznámky pod čarou

Příkazem

`\footnote{`*text poznámky pod čarou*`}`

lze vysázet poznámku pod čarou, tj. text vysázený v dolní části stránky oddělený od hlavního textu horizontální čarou a označený stejnou značkou jako odkaz na tuto poznámku v textu.

Začínající LaTeXisté[a] používají poznámek pod čarou velice často.

[a] LaTeXista je člověk užívající LaTeX

```
Začínající \LaTeX{isté}\footnote{%
\LaTeX{}ista je člověk užívající
\LaTeX} používají poznámek
pod čarou velice často.
```

2.8 Zvýraznění slov

V rukopisech psaných na psacím stroji se důležitá, zdůrazněná, slova <u>podtrhují</u>. V tištěných knihách se však tato slova *zvýrazňují*. Sazba se do *zvýrazněného* typu písma přepíná příkazem

`\emph{`*text*`}`

Argumentem tohoto příkazu je text, který má být *zvýrazněn*.

Užije-li se zvýraznění *v již zdůrazněném textu, potom LaTeX užije pro zdůraznění namísto kurzívy* vzpřímený *typ písma*

```
\emph{Užije-li se
  \emph{zvýraznění} v~již
  zdůrazněném textu, potom
  \LaTeX{} užije pro zdůraznění
  namísto kurzívy \emph{vzpřímený}
  typ písma}
```

2.9 Prostředí (Environments)

K označení zvláštních částí textu, které mají být vysázeny jinak než běžný text, slouží tzv. prostředí. Prostředí ve vstupním textu mají tvar:

`\begin{`*jméno*`}` *text* `\end{`*jméno*`}`

kde *jméno* je jméno daného prostředí. Platí, že prostředí se mohou vyskytovat jedno v druhém, přičemž je třeba dbát na správné pořadí. Jednotlivá prostředí se nemohou křížit.

```
\begin{aaa}...\begin{bbb}...\end{bbb}...\end{aaa}
```

V následujících oddílech budou vysvětlena všechna důležitá prostředí LaTeXu.

2.9.1 Výčty (itemize, enumerate a description)

Prostředí `itemize` je vhodné pro jednoduché výčty, prostředí `enumerate` pro číslované výčty a prostředí `description` pro popisné výčty.

1. Jednotlivé typy výčtů lze libovolně kombinovat:
   - Avšak výsledek pak může vypadat hrozně.
   - \- S pomlčkou.
2. Proto si pamatujte:

   Hloupé věci se nestanou moudřejšími, když jsou seřazeny do výčtu.

   Chytré věci však je dobré prezentovat právě ve výčtech.

```
\begin{enumerate}
\item Jednotlivé typy výčtů
lze libovolně kombinovat:
\begin{itemize}
\item Avšak výsledek pak
 může vypadat hrozně.
\item[-] S~pomlčkou.
\end{itemize}
\item Proto si pamatujte:
\begin{description}
\item[Hloupé] věci se nestanou
 moudřejšími, když jsou seřazeny
 do výčtu.
\item[Chytré] věci však je
  dobré prezentovat právě
  ve výčtech.
\end{description}
\end{enumerate}
```

2.9.2 Sazba na prapor a centrování (Flushleft, Flushright, Center)

V prostředí `flushleft` resp. `flushright` se sázený text doráží na levý resp. pravý okraj (tzv. „sazba na praporek“ — tj. bez zarovnávání okraje). Prostředí `center` slouží k centrování sazby. Pokud není použit příkaz \\, určí LaTeX řádkové zlomy sám.

Text zarovnaný
vlevo. LaTeX nezkouší zalomit
všechny řádky na stejnou délku.

```
\begin{flushleft}
Text zarovnaný \\ vlevo.
\LaTeX{} nezkouší zalomit
všechny řádky na stejnou délku.
\end{flushleft}
```

Text zarovnaný
vpravo. LaTeX nezkouší zalomit
všechny řádky na stejnou délku.

```
\begin{flushright}
Text zarovnaný \\ vpravo.
\LaTeX{} nezkouší zalomit
všechny řádky na stejnou délku.
\end{flushright}
```

V
ose tiskového
zrcadla.

```
\begin{center}
V\\ose tiskového\\zrcadla.
\end{center}
```

2.9.3 Citace (Quote, Quotation, Verse)

Prostředí `quote` se používá pro kratší citace, zdůrazněné věty a příklady.

Jedno typografické pravidlo pro délku řádku říká:

> *"No line should contain more than 66 characters."*
>
> „Žádný řádek by neměl obsahovat více než 66 písmen."

Důsledkem tohoto pravidla je to, že noviny se sází do více sloupců vedle sebe.

```
Jedno typografické pravidlo
pro délku řádku říká:
\begin{quote}
\emph{``No line should contain
more than 66~characters.''}

\uv{Žádný řádek by neměl
obsahovat více než 66 písmen.}
\end{quote}
Důsledkem tohoto pravidla je to,
že noviny se sází do více sloupců
vedle sebe.
```

Vedle prostředí `quote` existují další dvě prostředí: `quotation` a `verse`. Prostředí `quotation` je vhodné pro citace, které zahrnují několik odstavců, jelikož narozdíl od `quote` odsazuje `quotation` jednotlivé odstavce. Prostředí `verse` je užitečné pro sazbu poezie, kde je dělení na řádky předem známé. Jednotlivé verše se oddělují příkazem `\\` na konci řádku a jednotlivé sloky prázným řádkem mezi slokami.

Jára Cimrman se vždy snažil ukazovat svět takový jaký je se všemi krásami i ošklivostmi:

> Šel medvídek na procházku,
> uviděl tam sedmikrásku.
>
> Sedmikráska hezká,
> medvídek si pleská.
>
> *„Plesk, plesk. Plesk!“*
> Míšu zabil blesk!

Jak vidíme nebránil se tomu ani v tvorbě pro nejmenší.

```
Jára Cimrman se vždy snažil
ukazovat svět takový jaký je
se všemi krásami i ošklivostmi:
\begin{flushleft}
\begin{verse}
 Šel medvídek na procházku,\\
 uviděl tam sedmikrásku.

 Sedmikráska hezká,\\
 medvídek si pleská.

 \emph{\uv{Plesk, plesk. Plesk!}}\\
 Míšu zabil blesk!
\end{verse}
\end{flushleft}
Jak vidíme nebránil se tomu ani v~tvorbě
pro nejmenší.
```

2.9.4 Přímý výstup (verbatim, verb)

Text, který je uzavřen mezi `\begin{verbatim}` a `\end{verbatim}` bude vysázen přesně tak, jak byl zapsán ve zdrojovém souboru, tj. se všemi mezerami, tisknutelnými symboly, konci řádků a bez interpretace všech příkazů LaTeXu; tedy samozřejmě kromě `\end{verbatim}`. Toho lze s výhodou použít např. pro vytištění výpisů krátkých počítačových programů.

Uvnitř odstavce podobnou funkci plní příkaz

`\verb+`*text*`+`

Znak `+` je pouze příklad oddělovacího znaku. Jako oddělovací znak lze užít libovolný znak kromě písmen, znaků `*` a mezery. Většina příkladů příkazů LaTeXu v tomto dokumentu byla vysázena právě tímto příkazem.

Příkaz \ldots …

```
#include <stdio.h>

void main(void) {
  printf( "Hello World!\n");
}
```

```
Příkaz \verb|\ldots|  \ldots

\begin{verbatim}
#include <stdio.h>

void main(void) {
  printf( "Hello World!\n");
}
\end{verbatim}
```

```
Ohvězdičkovaná␣verze
prostředí␣␣␣␣␣verbatim
používá␣␣␣speciálního
symbolu␣(vaničky)␣␣pro
zvýraznění␣␣␣␣␣␣␣mezer.
```

```
\begin{verbatim*}
Ohvězdičkovaná verze
prostředí     verbatim
používá   speciálního
symbolu (vaničky)  pro
zvýraznění       mezer.
\end{verbatim*}
```

Příkaz `\verb` lze užít podobným způsobem, tj. v ohvězdičkované verzi:

```
jako␣␣␣␣␣␣zde␣:-)␣
```

```
\verb*|jako      zde :-) |
```

Prostředí `verbatim` a příkaz `\verb` se nesmí užívat v parametrech jiných příkazů.

2.9.5 Tabular

Prostředí `tabular` se užívá k sazbě tabulek. Jím vysázené tabulka může být vysázena spolu s volitelnými horizontálními a vertikálními linkami. LaTeX sám určí šířku sloupečků.

Argumentem *specifikace* příkazu

`\begin{tabular}{`*specifikace*`}`

se definuje formát tabulky. V argumentu *specifikace* užití `l` znamená sloupec s textem zarovnaným vlevo, `r` sloupec zarovnaný vpravo, `c` sloupec s centrovaným textem, `p{`*šířka*`}` sloupec zadané šířky s víceřádkovým textem (text se formátuje do odstavce) a `|` svislou čáru.

Uvnitř prostředí `tabular` znamená `&` skok na další sloupec tabulky, `\\` konec řádky tabulky, `\hline` (jako samostatný řádek) sází vodorovnou čáru v šířce tabulky.

7CE	hexadecimálně
3716	oktalově
11111001110	binárně
1998	decimálně

```
\begin{tabular}{|r|l|}
\hline
        7CE & hexadecimálně \\
       3716 & oktalově \\
11111001110 & binárně \\
\hline \hline
     1998   & decimálně \\
\hline
\end{tabular}
```

Vítejte v Krabičkově odstavci Věříme, že si zde užijete hodně pěkné podívané.

```
\begin{tabular}{|p{4.7cm}|}
\hline
Vítejte v~Krabičkově odstavci
Věříme, že si zde užijete
hodně pěkné podívané.\\
\hline
\end{tabular}
```

Pomocí konstrukce `@{...}` lze specifikovat oddělovač mezi danými sloupci. Tento příkaz potlačí sazbu mezisloupcových výplní (mezer) a namísto nich vysází to, co je vloženo mezi složené závorky konstrukce `@{...}`. Nejčastěji používaný případ použití této konstrukce, zarovnávání desetinných čísel, je uveden níže. Dalším případem užití je potlačení sazby mezisloupcových mezer vložením konstrukce `@{}`.

bez mezer kolem

```
\begin{tabular}{@{} l @{}}
\hline
bez mezer kolem\\
\hline
\end{tabular}
```

mezery vpravo i vlevo

```
\begin{tabular}{l}
\hline
mezery vpravo i vlevo\\
\hline
\end{tabular}
```

Jelikož LaTeX nemá vestavěnou podporu pro zarovnávání desetinných čísel v tabulkách na desetinnou čárku[5], je třeba užít švindl — použijeme dva sloupce: vpravo zarovnaný sloupec pro celočíselnou část a vlevo zarovnaný sloupec pro část desetinnou. Příkaz `@{,}` nahradí mezisloupcové výplně symbolem pro desetinnou čárku. Při sazbě pak nesmíme zapomenout nahradit desetinnou čárku odělovačem sloupců, tedy symbolem `&`. Popis takového zdvojeného sloupce lze vytvořit pomocí příkazu `\multicolumn`.

výraz s π	Hodnota
π	3,1416
π^π	36,46
$(\pi^\pi)^\pi$	80662,7

```
\begin{tabular}{c r @{,} l}
výraz s~$\pi$      &
\multicolumn{2}{c}{Hodnota} \\
\hline
$\pi$               & 3&1416  \\
$\pi^{\pi}$         & 36&46   \\
$(\pi^{\pi})^{\pi}$ & 80662&7 \\
\end{tabular}
```

[5] Je-li v systému naistalován soubor balíků „tools“ lze užít balíku dcolumn.

2.10 Plovoucí objekty (Floating Bodies)

V dnešní době většina publikací obsahuje množství různých obrázků a tabulek. Tyto objekty vyžadují speciální zpracování, jelikož je většinou nelze rozdělit na několik stránek. Jednou z metod by mohlo být vždy při sazbě obrázku či tabulky, které se již nevejdou na aktuálně sázenou stránku začít stránku novou. Tento přístup by však vedl k tomu, že by některé stránky zůstávaly poloprázné což by jistě nevypadalo moc pěkně.

Vhodným řešením tohoto problému je to, že se obrázek či tabulka, které se již na aktuálně sázenou stránku nevejdou, prostě na této stránce nevysází a z jejich sazbou se posečká až na další vhodné místo na některé z následujích stránek. LaTeX pro takové plovoucí objekty nabízí dvě prostředí. Jedno pro obrázky a druhé pro tabulky. K tomu abychom mohli plně využít těchto prostředí je potřeba alespoň přibližně porozumnět jak LaTeX vnitřně s těmito prostředími pracuje. Jinak v nás tato prostředí neustále budou vyvolávat pocity marnosti, jelikož nikdy nevysází náš obrázek či tabulku na místo, kam bychom to zrovna potřebovali.

Nejprve se podívejme na to jak vypadají příkazy pro práci s plovoucími objekty:

Libovolný materiál uzavřený v prostředí `figure` či `table` je považován za plovoucí objekt. Obě prostředí mají ještě nepovinný parametr *specifikace umístění*

`\begin{figure}[`*specifikace umístění*`]` či `\begin{table}[`*specifikace umístění*`]`

Tento parametr řekne LaTeXu kam na stránce lze daný plovoucí objekt umístit. Argument *specifikace umístění* je řetězec sestavený ze znaků *povolení umístění*. Viz tabulku 2.2.

Tabulka 2.2: Znaky povolení umístění

znak	povolené místo umístění ...
`h`	*zde* (here) právě v tom místě textu, kde se toto prostředí objevilo. To je vhodné zejména pro malé objekty.
`t`	*v horní části* (top) stránky
`b`	*v dolní části* (bottom) stránky
`p`	na zvláštní *stránce* (page), na níž budou pouze plovoucí objekty.
!	bez toho aniž by se bral ohled na většinu vnitřních parametrů prostředí[a], které by mohly zabránit umístění objektu v daném místě.

[a] Jako například maximální dovolený počet plouvoucích objektů na jedné stránce

Tabulku lze umístit například v prostředí, které začíná např. takto:

```
\begin{table}[!hbp]
```

Parametr specifikace umístění `[!hbp]` dovoluje LaTeXu umístit tabulku buď právě zde (`h`) nebo ve spodní části (`b`) některé, třeba následující, stránky nebo na zvláštní stránce určené pouze pro plovoucí objekty (`p`) a to vše dokonce i v případě, že to nebude podle všech pravidel sazby (`!`). Jestliže umístění není specifikováno pomocí parametru, pak standardní třídy předpokládají `[tbp]`.

LaTeX bude umisťovat každý plovoucí objekt, na nějž narazí podle specifikace autora. Nemůže-li být plovoucí objekt umístěn na aktuální stránce, je umístěn do fronty[6] obrázků či tabulek. Na počátku každé nové stránky LaTeX nejprve zkouší, zda nemá frontu natolik plnou, aby z objektů v ní umístěných mohl vysázet zvláštní stránku. Není-li možné vysázet takovou stránku, první objekt v každé frontě se zpracovává tak, jakoby se vyskytly ve zdrojovém textu právě na této stránce, tj. LaTeX zkouší znovu umístit objekt na danou stránku podle specifikace *specifikace umístění*. Samozřejmě se již nezpracovává požadavek na umístění `h`, který již pozbyl platnosti. Jakýkoli další objekt, jež se objeví ve zdrojovém textu se zařadí na konec příslušné fronty. LaTeX při sazbě přesně dodržuje původní pořadí daného typu objektů. Proto se může stát, že jeden objekt, který nelze nikde umístit pozdrží sazbu všech následujích objektů třeba až na úplný konec dokumentu. Proto:

> Neumístí-li LaTeX objekt, tj. obrázek či tabulku, na autorem očekávané místo, tak pravděpodobně některý z předchozích objektů ucpal příslušnou frontu.

Pro práci z prostředím `table` a `figure` je dobré znát i některé další věci o nichž se zmíníme nyní.

Pomocí příkazu

`\caption{`*text popisu*`}`

lze nadefinovat popisný titulek pro daný plovoucí objekt. Popis objektu se vedle *textu popisu* skládá ještě z čísla a řetězce „Obrázek“ nebo „Tabulka“, jež přidá LaTeX automaticky. Číselné řady pro číslování obrázků a tabulek má ve správě rovněž LaTeX sám.

Dva příkazy

`\listoffigures` a `\listoftables`

pracují obdobně jako příkaz `\tableofcontents`, tj. vysází seznam obrázků resp. tabulek. V těchto seznamech je zopakován úplný *text popisu*. Mnozí

[6] FIFO — kdo přijde první bude první vysázen

autoři mají však sklon k delším popiskům, proto je třeba do seznamu užít zkrácené formy popisu čehož lze dosáhnout obdobně jako u příkazů pro strukturalizaci textu zadáním zkrácené fromy popisku jako nepovinný parametr těsně za příkaz `\caption`.

```
\caption[Krátký popisek]{Velmi dlouhý popisek nějakého pěkného objektu}
```

Užitím příkazů `\label` a `\ref` lze tvořit odkazy uvnitř textu na příslušné obrázky resp. tabulku.

V následujícím příkladu je vykreslen prázdný čtverec, který je jako obrázek vložen do textu. Toho lze využít například v případě, že potřebujete vynechat ve výsledném dokumentu místo pro vlepení obrázku (např. fotografie).

```
Na obrázku~\ref{obrazek} vidíte uspořádání experimentu.
\begin{figure}[!hbp]
\makebox[\textwidth]{\framebox[5cm]{\rule{0pt}{5cm}}}
\caption{Uspořádání experimentu} \label{obrazek}
\end{figure}
```

V předchozím příkladu[7] se LaTeX *bez skrupulí* (`!`) pokusí umístit obrázek *zde* (`h`). Nebude-li to možné, pokusí se umístit obrázek na spodní okraj stránky (`b`). Nepodaří-li se mu vysázet obrázek na aktuální stránku zváží zda není možné umístit obrázek na zvláštní stránku spolu s nevysázenými tabulkami z fronty tabulek. Není-li dost materiálu k tomu, aby bylo možné takovou stránku vysázet začne LaTeX sázet novou stránku a znovu se pokusí umístit obrázek tak, jako kdyby se jeho popis objevil ve zdrojovém textu právě nyní.

V některých případech je potřeba užít příkaz

`\clearpage` či dokonce `\cleardoublepage`

kterým se LaTeXu nařídí, aby okamžitě umístil všechny plovoucí objekty z front obrázků a tabulek a poté začal sázet novou stránku. Pomocí příkazu `\cleardoublepage` se navíc nařídí pokračování další sazby až na další levé stránce.

2.11 Přidání nových příkazů a prostředí

V první kapitole je napsáno, že LaTeX k tomu, aby dokázal vysázet dokument daného typu, potřebuje pouze informace o logické struktuře tohoto dokumentu. Toto je základní myšlenka LaTeXu, ale v praxi se dostáváme do

[7] za předpokladu, že fronta obrázků je prázdná

situací, kdy LaTeX nenabízí příkaz či prostředí, které by splňovalo autorův požadavek na strukturalizaci dokumentu.

Jedním z možných řešení, je užít několika příkazů LaTeXu a uspořádat si sazbu dle potřeby. To není takový problém, je-li to potřeba provést pouze jednou, avšak pokud je potřeba danou složitější konstrukci příkazů ve zdrojovém textu nekolikrát opakovat je tento přístup nevhodný. Nevhodnost tohoto přístupu se projeví zejména tehdy, když je třeba tuto opakovanou složitější konstrukci upravit, potom je třeba procházet celý zdrojový text a editovat všechny výskyty dané konstrukce.

K řešení takových problémů LaTeX nabízí možnost definice vlastních příkazů a prostředí.

2.11.1 Nové příkazy

Nový příkaz lze vytvořit pomocí příkazu:

`\newcommand{`*jméno*`}[`*číslo*`]{`*definice*`}`

Při běžném užití příkazu stačí dva argumenty: *jméno* nového příkazu, který právě tvoříme a jeho *definice*. Argument *číslo* v hranatých závorkách je volitelný a užívá se v případě, když chceme aby nový příkaz sám používal argumenty. Těchto argumentů může být maximálně devět, proto argument *číslo* může nabývat hodnot 1–9.

Následující dva příklady ilustrují myšlenku užití příkazu `\newcommand`. V první příkladu je nadefinován nový příkaz `\uvodlat`, jež slouží jako zkratka pro „Ne příliš stručný úvod do systému LaTeX 2_ε“. Takový příkaz je vhodný v případě, když se několikrát v textu opakuje stále totéž slovní spojení, zde název této příručky.

„Ne příliš stručný úvod do systému LaTeX 2_ε“ … „Ne příliš stručný úvod do systému LaTeX 2_ε“

```
\newcommand{\uvodlat}
    {Ne příliš stručný úvod
                do systému \LaTeXe}
% následující se objeví v těle TeXtu
\uv{\uvodlat} \ldots{} \uv{\uvodlat}
```

Další příklad ilustruje užití argumentu *číslo*. Konstrukce `#`n je nahrazena n-tým argumentem nového příkazu, přičemž jak již bylo uvedeno $n \in \{1 \ldots 9\}$. Tedy v našem případě `#1` bude nahrazena prvním argumentem příkazu `\uvlat`.

- *Ne příliš* stručný úvod do systému LaTeX 2_ε
- *Velmi* stručný úvod do systému LaTeX 2_ε

```
\newcommand{\uvlat}[1]
   {\emph{#1} stručný úvod
            do systému \LaTeXe}
% následující se objeví v těle TeXtu
\begin{itemize}
\item \uvlat{Ne příliš}
\item \uvlat{Velmi}
\end{itemize}
```

LaTeX nedovolí vytvořit nový příkaz se jménem již existujícího příkazu. Jestliže je třeba přepsat definici některého příkazu je třeba užít příkazu `\renewcommand`, který má syntax stejnou jako příkaz `\newcommand`. V některých případech lze užít též užít příkazu `\providecommand`, který dělá totéž co `\newcommand`, avšak v případě, že příkaz tohoto jména je již definován, LaTeX tuto novou definici ignoruje.

2.11.2 Nové prostředí

Obdobně jako je možné vytvářet nové příkazy je možné tvořit i vlastní nová prostředí. Například pro psaní této příručky její autor, Tobias Oetiker, vytvořil několik zvláštních prostředí pro často se vyskytující struktury, jako příklady, části kódu, rámečky s definicí příkazu...

Nové prostředí se definuje příkazem `\newenvironment`, jež má následující syntax:

`\newenvironment{`*jméno*`}[`*číslo*`]{`*před*`}{`*po*`}`

Stejně jako u příkazu `\newcommand` i u příkazu `\newenvironment` lze zadat nepovinný parametr *číslo* specifikující počet argumentů ovlivňujících nějakým způsobem vytvářené prostředí. Příkazy, které se objeví v části *před* se provedou před tím, než se začne zpracovávat text v prostředí uzavřený. Příkazy obsažené v části *po* se začnou vykonávat v okamžiku, kdy LaTeX začne zpracovávat příkaz `\end{`*jméno*`}`.

Následující příklad ilustruje užití příkazu `\newenvironment`.

> Nebude tvář lidu jasná ...

```
\newenvironment{citace}
    {\begin{quote}}{\end{quote}}
% následující se objeví v těle TeXtu
\begin{citace}
Nebude tvář lidu jasná \ldots
\end{citace}
```

Obdobně jako při definování nových příkazů LaTeX dává pozor na to, aby nebylo nadefinováno prostředí se stejným jménem, jako má některé již existující. Je-li třeba předefinovat některé prostředí užívá se k tomu příkaz `\renewenvironment`, jež má stejnou syntax jako příkaz `\newenvironment`.

Kapitola 3

Sázení matematických vzorců

Tak jste dospěli až do bodu, kdy se budeme moci dotknout nejsilnější stránky TEXu, sazby matematiky. Je však třeba podotknout, že se v této kapitole dotkneme věci pouze povrchně, více o pravidlech sazby matematiky v TEXu se můžete dočíst v [2]. Obsah této kapitoly bude pravděpodobně dostačovat většině čtenářů k tomu, aby mohli začít psát odborné texty, obsahující matematiku. Nenaleznete-li v této kapitole některou matematickou strukturu, jež byste potřebovali vysázet v této kapitole, pravděpodobně naleznete vhodný prostředek v AMS-LaTEXu[1] či nějakém jiném balíku maker, zaměřeným na sazbu matematiky.

3.1 Obecný úvod

TEX používá pro sazbu matematiky zvláštního módu. Matematické části textu uvnitř odstavce se uzavírají mezi `\(` a `\)`, nebo mezi `$` a `$` či mezi `\begin{math}` a `\end{math}`.

Nechť a a b jsou odvěsny a c přepona, pak platí $c^2 = a^2 + b^2$ (Pythagorova věta).

```
Nechť $a$ a $b$ jsou
odvěsny a $c$ přepona,
pak platí $c^{2}=a^{2}+b^{2}$
(Pythagorova věta).
```

TEX se vyslovuje jako $\tau\epsilon\chi$.

100 m^2 obytné plochy.

I $\heartsuit$ TEX.

```
\TeX\ se vyslovuje jako
 $\tau\epsilon\chi$.\\[6pt]
100~m$^{2}$ obytné plochy.\\[6pt]
I $\heartsuit$ \TeX.
```

Rozsáhlejší matematické vzorce či rovnice, je lépe sázet na zvláštní řádky. V tom případě se píší mezi `\[` a `\]` či mezi `\begin{displaymath}` a `\end{displaymath}`. Takto vysázené rovnice však nejsou číslovány. Pokud rovnice uzavřeme do

[1] `CTAN:/tex-archive/macros/latex/packages/amslatex`

prostředí `equation` LaTeX, vedle odsazení matematiky na zvláštní řádek bude rovnice navíc číslovaná.

Nechť a a b jsou odvěsny a c přepona, pak platí

$$c^2 = a^2 + b^2$$

(Pythagorova věta)

```
Nechť $a$ a $b$ jsou
odvěsny a $c$ přepona,
pak platí
\begin{displaymath}
c^{2}=a^{2}+b^{2}
\end{displaymath}
(Pythagorova věta)
```

Pomocí již známých příkazů `\label` a `\ref` se lze v textu odkazovat na takto očíslované rovnice.

$$\epsilon > 0 \tag{3.1}$$

Z (3.1) dostáváme……

```
\begin{equation} \label{eq:eps}
\epsilon > 0
\end{equation}
Z~(\ref{eq:eps}) dostáváme\ldots
\ldots
```

V následujícím příkladu si všimněte , že výrazy se jinak vysází uvnitř odstavce a jinak na zvláštním řádku

$\lim_{n \to \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$

$$\lim_{n \to \infty} \sum_{k=1}^n \frac{1}{k^2} = \frac{\pi^2}{6}$$

```
$\lim_{n \to \infty}
\sum_{k=1}^n \frac{1}{k^2}
= \frac{\pi^2}{6}$
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
\begin{displaymath}
\lim_{n \to \infty}
\sum_{k=1}^n \frac{1}{k^2}
= \frac{\pi^2}{6}
\end{displaymath}
```

Sazba v matematickém režimu (math mode) se liší od sazby v textovém režimu (text mode) zvláště v těchto bodech:

1. Mezery a ukončení řádku ve vstupním textu nemají žádný význam, neboť vše se sází podle pravidel matematické sazby, případně se mezery zadávají speciálními příkazy jako `\,`, `\quad`, nebo `\qquad`.

   $$\forall x \in \mathbf{R}: \qquad x^{2} \geq 0 \tag{3.2}$$

   ```
   \begin{equation}
     \forall x \in \mathbf{R}:
     \qquad x^{2} \geq 0
   \end{equation}
   ```

2. Prázdné řádky jsou zakázány (tj. matematický vzorec musí být v jednom odstavci).

3. Každé písmeno je považováno za jméno proměnné a je vysázeno odpovídajícím způsobem (tzv. matematická kurzíva). Chceme-li uvnitř matematického textu sázet normální text, musíme jej uzavřít do `\textrm{...}`.

$$x^2 \geq 0 \quad \textrm{pro všechna } x \in \mathbf{R} \tag{3.3}$$

```
\begin{equation}
 x^{2} \geq 0\quad
  \textrm{pro všechna }
 x\in\mathbf{R}
\end{equation}
```

Matmetici nadělají mnoho povyku kolem symbolů, jež jsou při sazbě užity. V předchozím příkladu by možná pro vyznačení oboru reálných čísel bylo lepší použít zdvojené písmo, které lze zízka užitím příkazu `\mathbb` z balíku amsfonts či amssymb. Předchozí příklad bude tedy mít formu:

$$x^2 \geq 0 \quad \textrm{pro všechna } x \in \mathbb{R}$$

```
\begin{displaymath}
x^{2} \geq 0\quad
\textrm{pro všechna }x\in\mathbb{R}
\end{displaymath}
```

3.2 Tvorba skupin v matematickém režimu

Většina příkazů v matematickém reřimu má vliv pouze na bezprostředně následující znak. Je-li třeba, aby příkaz ovlivnil sazbu více znaků, je z nich potřeba, za pomoci složených závorek `{...}`, vytvořit skupinu.

$$a^x + y \neq a^{x+y} \tag{3.4}$$

```
\begin{equation}
a^x+y \neq a^{x+y}
\end{equation}
```

3.3 Objekty matematických vzorců

V této kapitole jsou stručně popsány nejdůležitější objekty, které se používají v matematických vzorcích. Seznam všech použitelných symbolů obsahuje oddíl 3.10 na straně 46.

Malá písmena řecké abecedy se sází pomocí příkazů `\alpha`, `\beta`, `\gamma`, ..., velká písmena[2] se sází příkazy `\Gamma`, `\Delta`, ...

$\lambda, \xi, \pi, \mu, \Phi, \Omega$

```
$\lambda,\xi,\pi,\mu,\Phi,\Omega$
```

[2] LaTeX nemá nadefinován příkaz pro velké písmeno Apfa, jelikož vypadá stejně jako velké mísmeno A.

Exponenty a indexy se zadávají pomocí znaků ^ a _.

$a_1 \qquad x^2 \qquad e^{-\alpha t} \qquad a^3_{ij}$
$e^{x^2} \neq {e^x}^2$

```
$a_{1}$ \qquad $x^{2}$ \qquad
$e^{-\alpha t}$ \qquad
$a^{3}_{ij}$\\
$e^{x^2} \neq {e^x}^2$
```

Značka pro **odmocninu** se sází pomocí příkazu `\sqrt`, n^{th}-tá odmocnina se genuje pomocí `\sqrt[`n`]`. Velikost odmocnítka LaTeX volí automaticky. Je-li třeba vysázet pouze symbol odmocniny, bez čárky nad odmocněncem, užijeme `\surd`.

$\sqrt{x} \qquad \sqrt{x^{2}+\sqrt{y}} \qquad \sqrt[3]{2}$
$\surd[x^2 + y^2]$

```
$\sqrt{x}$ \qquad
$\sqrt{ x^{2}+\sqrt{y} }$
\qquad $\sqrt[3]{2}$\\[3pt]
$\surd[x^2 + y^2]$
```

Příkazy `\overline` či `\underline` zajistí sazbu **horizontální linky** nad či pod požadovanou částí matematického výrazu.

$\overline{m+n}$

```
$\overline{m+n}$
```

Příkaz `\overbrace` resp. `\underbrace` vysází vodorovnou **složenou závorku** nad resp. pod požadovanou částí matematického výrazu.

$\underbrace{ a+b+\cdots+z }_{26}$

```
$\underbrace{ a+b+\cdots+z }_{26}$
```

Příkazy pro sázení matematických **akcentů** jako stříška nad proměnnou, nebo vlnovka nad proměnnou, obsahuje tabulka 3.1. Delší vlnovky a stříšky, které mohou být nad více znaky, dostaneme pomocí `\widetilde` a `\widehat`. Označení derivace se zadává pomocí znaku apostrof `'`.

$$y=x^{2}\qquad y'=2x\qquad y''=2$$

```
\begin{displaymath}
y=x^{2}\qquad y'=2x\qquad y''=2
\end{displaymath}
```

Vektory jsou velice často označovány malou šipkou nad proměnnou. Pro tento účel je definován příkaz `\vec`. K označení vektoru z A do B lze užít dvou příkazů `\overrightarrow` a `\overleftarrow`.

$$\vec a\quad\overrightarrow{AB}$$

```
\begin{displaymath}
\vec a\quad\overrightarrow{AB}
\end{displaymath}
```

Matematické **funkce** se v literatuře obvykle nesází kurzívou (jako jména proměnných), nýbrž „normálním“ vzpřímeným typem písma. K tomu slouží následující příkazy:

`\arccos`	`\cos`	`\csc`	`\exp`	`\ker`	`\limsup`	`\min`	`\sinh`
`\arcsin`	`\cosh`	`\deg`	`\gcd`	`\lg`	`\ln`	`\Pr`	`\sup`
`\arctan`	`\cot`	`\det`	`\hom`	`\lim`	`\log`	`\sec`	`\tan`
`\arg`	`\coth`	`\dim`	`\inf`	`\liminf`	`\max`	`\sin`	`\tanh`

$$\lim_{n\rightarrow 0}\frac{\sin x}{x}=1$$

```
\[\lim_{n \rightarrow 0}
\frac{\sin x}{x}=1\]
```

Pro funkci modulo existují dva různé příkazy: `\bmod` pro binární operátor $a \bmod b$ a `\pmod...` pro zadání ve formě $x \equiv a \pmod{b}$.

V české technické literatuře se však některé funkce, např. „tangens“ apod., označují jinak než v americké. Tyto lze v LaTeXu v případě potřeby snadno dodefinovat. V následujícím příkladu je ilustrováno, jak lze pomocí příkazu `\mathop` nadefinovat označení pro novou funkci, v našem případě `\tg` pro označení funkce tangents.

$\mathop{\rm tg}\nolimits\alpha\ne1$

```
\newcommand{\tg}{\mathop{\rm tg}\nolimits}
%%%%%%%%
$\tg{\alpha}\ne1$
```

Zlomek (angl. fraction) se vysází příkazem `\frac{...}{...}`. Pro jednoduché zlomky, např. 1/2, lze však použít také operátor `/`.

$1\frac{1}{2}$ koláče

$$\frac{x^{2}}{k+1}\qquad x^{\frac{2}{k+1}}\qquad x^{1/2}$$

```
$1\frac{1}{2}$~koláče
\begin{displaymath}
\frac{ x^{2} }{ k+1 }\qquad
x^{ \frac{2}{k+1} }\qquad
x^{ 1/2 }
\end{displaymath}
```

Binomické koeficienty či jiné obdobné struktury lze sázet buď pomocí příkazu `{... \choose ...}` či pomocí příkazu `{... \atop ...}`. Druhý příkaz se chová obdobně jako první, pouze s tím rozdílem, že nevysází závorky.

$$\binom{n}{k}\qquad {x \atop y+2}$$

```
\begin{displaymath}
{n \choose k}\qquad {x \atop y+2}
\end{displaymath}
```

Značka **integrálu** se sází příkazem `\int`, značka **sumy** příkazem `\sum`. Horní a dolní mez se zadává pomocí `^` a `_` (jakoby horní a dolní index).

$$\sum_{i=1}^{n} \qquad \int_{0}^{\frac{\pi}{2}} \qquad$$

```
\begin{displaymath}
\sum_{i=1}^{n} \qquad
\int_{0}^{\frac{\pi}{2}} \qquad
\end{displaymath}
```

Meze integrálu se obvykle sázejí *vedle* značky integrálu (,aby se šetřilo místem). Pokud požadujeme meze nad a pod značkou, použijeme příkaz `\limits`. Naproti tomu meze u sumy jsou sázeny nad a pod značku, pouze u vzorců v textu, nebo použitím příkazu `\nolimits`, se meze vysázejí vedle značky.

$$\sum\nolimits_{i=1}^{n} \qquad \int\limits_{0}^{\frac{\pi}{2}} \qquad$$

```
\begin{displaymath}
\sum\nolimits_{i=1}^{n} \qquad
\int\limits_{0}^{\frac{\pi}{2}} \qquad
\end{displaymath}
```

Pro **závorky** a jiné ohraničující elementy obsahuje TEX mnoho různých symbolů (např. $[\ \langle\ \|\ \updownarrow$). Kulaté a hranaté závorky se zadávají odpovídajícími znaky, složené závorky pomocí `\{` a `\}`, ostatní speciálními příkazy (např. `\updownarrow`). Seznam všech druhů „závorek“ naleznete v tabulce 3.8 na straně 48.

$$a, b, c \neq \{a, b, c\}$$

```
\begin{displaymath}
{a,b,c}\neq\{a,b,c\}
\end{displaymath}
```

Uvedeme-li před otevírací závorkou příkaz `\left` a před odpovídající zavírací závorkou příkaz `\right`, LaTeX automaticky zvolí velikost závorek. Zde je nutno upozornit, že každému příkazu `\left` musí odpovídat `\right`, i když bychom nechtěli uzavírací závorku sázet. Pak se používá příkaz `\right` s tečkou (tj. „`\right.`“).

$$1 + \left(\frac{1}{1-x^{2}} \right)^3$$

```
\begin{displaymath}
1 + \left( \frac{1}{ 1-x^{2} }
    \right) ^3
\end{displaymath}
```

V některých případech je vhodné velikost závorek nastavit „ručně“. K tomu slouží příkazy `\big`, `\Big`, `\bigg` a `\Bigg`, jako prefix (předponu) většiny závorkovacích příkazů[3].

[3]Tyto příkazy v případě, že se změní velikost písma či v případě užití globálního parametru `11pt` či `12pt`, nepracují tak, jak by se od nich očekávalo. Tuto vlastnost však napravuje balík exscale nebo amsmath

$\Big((x+1)(x-1) \Big)^{2}$

$\big(\Big(\bigg(\Bigg(\quad \big\}\Big\}\bigg\}\Bigg\}\quad \big\|\Big\|\bigg\|\Bigg\|$

```
$\Big( (x+1) (x-1) \Big) ^{2}$\\
$\big(\Big(\bigg(\Bigg($\quad
$\big\}\Big\}\bigg\}\Bigg\}$\quad
$\big\|\Big\|\bigg\|\Bigg\|$
```

Tři tečky se ve vzorcích zadávají příkazy `\ldots` a `\cdots`. Příkaz `\ldots` sází tečky na základní čáru (low), `\cdots` sází tečky doprostřed výšky znaků (centered). Kromě toho existuje příkaz `\vdots` pro vertikální a `\ddots` pro diagonální tečky. Další příklady naleznete v oddílu 3.5.

$$x_{1},\ldots,x_{n} \qquad x_{1}+\cdots+x_{n}$$

```
\begin{displaymath}
x_{1},\ldots,x_{n} \qquad
x_{1}+\cdots+x_{n}
\end{displaymath}
```

Příkaz `\stackrel` se dvěma argumenty vysází jeden symbol nad druhý. První argument je sázen menší velikostí.

$$\vec{x} \stackrel{\mathrm{def}}{=} (x_{1},\ldots,x_{n})$$

```
\begin{displaymath}
\vec{x}
\stackrel{\mathrm{def}}{=}
(x_{1},\ldots,x_{n})
\end{displaymath}
```

V matematickém režimu lze také příkazem `\cal` přepnout implicitní matematickou kurzívu na psané písmo (kaligrafické písmo). Můžeme pak ovšem použít pouze velká písmena.

$$z \in Z \quad z \in {\cal Z}$$

```
\begin{displaymath}
z \in Z \quad
z \in {\cal Z}
\end{displaymath}
```

3.4 Mezery v matematice

Nejsme-li spokojeni ve vzorcích s **mezerami**, které zvolil TeX, můžeme je změnit explicitními příkazy. Mezi nejdůležitější patří `\,` pro malou mezeru, `\␣` pro střední (`␣` nahrazuje znak mezera), `\quad` a `\qquad` pro velké mezery. Příkaz `\!` mezeru zmenšuje (záporná malá mezera).

$$F_{n} = F_{n-1} + F_{n-2} \qquad n \ge 2$$

```
\begin{displaymath}
F_{n} = F_{n-1} + F_{n-2}
 \qquad n \ge 2
\end{displaymath}
```

$$\int\!\!\!\int_D g(x,y)\,\mathrm{d}x\,\mathrm{d}y$$

namísto

$$\int\int_D g(x,y)\mathrm{d}x\mathrm{d}y$$

```
\newcommand{\ud}{\mathrm{d}}
\begin{displaymath}
\int\!\!\!\int_{D} g(x,y)
  \, \ud x\, \ud y
\end{displaymath}
namísto
\begin{displaymath}
\int\int_{D} g(x,y)\ud x \ud y
\end{displaymath}
```

Znak „d“ v diferenciálu se obvykle sází vzpřímeným typem písma.

3.5 Sazba na více řádků

Pro sazbu **matic** aj. se používá prostředí `array`, které se chová podobně jako prostředí `tabular`. Příkazem `\\` se dělí řádky.

$$\mathbf{X} = \left(\begin{array}{ccc} x_{11} & x_{12} & \ldots \\ x_{21} & x_{22} & \ldots \\ \vdots & \vdots & \ddots \end{array} \right)$$

```
\begin{displaymath}
\mathbf{X} =
\left( \begin{array}{ccc}
x_{11} & x_{12} & \ldots \\
x_{21} & x_{22} & \ldots \\
\vdots & \vdots & \ddots
\end{array} \right)
\end{displaymath}
```

Prostředí `array` lze rovněž použít pro sazbu výrazů, jež obsahují oddělovač v tečkované formě, jako neviditelnou pravou závorku:

$$y = \left\{ \begin{array}{ll} a & \textrm{if } d>c \\ b+x & \textrm{in the morning} \\ l & \textrm{all day long} \end{array} \right. \tag{3.5}$$

```
\begin{equation}
y = \left\{ \begin{array}{ll}
 a & \textrm{if $d>c$}\\
 b+x & \textrm{in the morning}\\
 l & \textrm{all day long}
  \end{array} \right.
\end{equation}
```

Víceřádkové vzorce nebo systémy rovnic se sázejí pomocí prostředí `eqnarray` a `eqnarray*` místo `equation`. Při použití `eqnarray` obsahuje každý řádek číslo rovnice (pokud to u některého řádku nezakážeme), při `eqnarray*` nikoli (jako v `displaymath`). Systémy rovnic, které jsou číslovány jako jeden objekt, se mohou zadávat v prostředí `array` uvnitř prostředí `equation`, viz předchozí příklad.

Na prostředí `eqnarray` a `eqnarray*` lze nahlížet jako na tabulku o třech sloupcích se specifikací `{rcl}`, kde prostřední sloupec lze užít pro některý z operátorů; většinou rovnítko. Příkaz \\ i zde slouží k oddělování jednotlivých řádek.

$$\begin{array}{rcl} f(x) & = & \cos x \quad (3.6) \\ f'(x) & = & -\sin x \quad (3.7) \\ \int_{0}^{x} f(y)dy & = & \sin x \quad (3.8) \end{array}$$

```
\begin{eqnarray}
f(x) & = & \cos x     \\
f'(x) & = & -\sin x   \\
\int_{0}^{x} f(y)dy &
 = & \sin x
\end{eqnarray}
```

Všimněte si, že kolem středního sloupce je zbytečně mnoho místa. Totom místo lze měnit nastavením `\setlength\arraycolsep{2pt}` tak, jako v následujícím příkladu.

Dlouhé rovnice nejsou automaticky LaTeXem děleny na několik řádek. Autor textu proto musí určit, kde se rovnice má rozdělit a musí vložit patřičnou mezeru na počátek druhého řádku rozdělené rovnice. Následující dva příklady ilustrují dva nejčastěji užívané přístupy k řešení tohoto problému.

$$\begin{array}{rcl} \sin x & = & x - \frac{x^{3}}{3!} + \frac{x^{5}}{5!} - \\ & & {} - \frac{x^{7}}{7!} + \cdots \quad (3.9) \end{array}$$

```
{\setlength\arraycolsep{2pt}
\begin{eqnarray}
\sin x & = & x -\frac{x^{3}}{3!}
     +\frac{x^{5}}{5!}-{}
                    \nonumber\\
 & & {}-\frac{x^{7}}{7!}+{}\cdots
\end{eqnarray}}
```

$$\begin{array}{l} \cos x = 1 - \frac{x^{2}}{2!} + \\ \qquad + \frac{x^{4}}{4!} - \frac{x^{6}}{6!} + \cdots \quad (3.10) \end{array}$$

```
\begin{eqnarray}
\lefteqn{ \cos x = 1
     -\frac{x^{2}}{2!} +{} }
                    \nonumber\\
 & & {}+\frac{x^{4}}{4!}
     -\frac{x^{6}}{6!}+{}\cdots
\end{eqnarray}
```

Příkaz `\nonumber` zajistí, že pro daný řádek se nesází číslo rovnice.

Někdy je velice těžké, aby vertikálně zarovnané rovnice za použití těchto dvou způsobů vypadali hezky. V takovém případě je dobré užít balíku amsmath, který k řešení takových problémů nabízí daleko silnější aparát.

3.6 Velikost písma při sazbě matematiky

V matematickém režimu velikost písma vybírá TeX podle kontextu. Například indexy jsou sazeny menším typem písma. Jestliže je třeba do sázeného matematického vzorce vysázet text v běžném vzpřímeném typu písma, je možné užít příkaz `\textrm`, mechanismus automatické změny velikosti písma dle kontextu však přestane fungovat, jelikož příkaz `\textrm` dočasně přepne sazbu do textového režimu. Chceme-li, aby mechanismus automatické změny velikosti nepřestal fungovat, je lépe užít namísto příkazu `\textrm` příkaz `\mathrm`.

$$2^{\textrm{nd}} \quad 2^{\mathrm{nd}} \tag{3.11}$$

```
\begin{equation}
2^\textrm{nd} \quad
2^\mathrm{nd}
\end{equation}
```

Někdy je však LaTeXu potřeba říci jakou velikost fontu má užít. V matematickém režimu se velikost fontu nastavuje těmito čtyřmi příkazy:

`\displaystyle` ($\displaystyle 123$), `\textstyle` ($\textstyle 123$), `\scriptstyle` ($\scriptstyle 123$) a `\scriptscriptstyle` ($\scriptscriptstyle 123$).

Změnou matematického stylu je ovlivněna také sazba limit některých matematických symbolů.

$$\mathop{\mathrm{corr}}(X,Y)= \frac{\displaystyle \sum_{i=1}^n(x_i-\overline x)(y_i-\overline y)}{\displaystyle\biggl[\sum_{i=1}^n(x_i-\overline x)^2 \sum_{i=1}^n(y_i-\overline y)^2 \biggr]^{1/2}}$$

```
\begin{displaymath}
\mathop{\mathrm{corr}}(X,Y)=
 \frac{\displaystyle
   \sum_{i=1}^n(x_i-\overline x)
   (y_i-\overline y)}
  {\displaystyle\biggl[
 \sum_{i=1}^n(x_i-\overline x)^2
\sum_{i=1}^n(y_i-\overline y)^2
\biggr]^{1/2}}
\end{displaymath}
```

Toto je jeden z případů, kdy je třeba použít menších závorek, než nabízí standardní `\left[` a `\right]`.

3.7 Popis proměnných

Po některých matematických výrazech je nutné uvést odstavec popisující proměnné užité ve výrazu. Následující příklad ukazuje jeden způsob možného přístupu k tomuto problému.

$$a^2 + b^2 = c^2$$

Kde: a, b – jsou přilehlé odvěsny pravoúhlého trojúhelníku.

c – je přepona pravoúhlého trojúhelníku.

```
\begin{displaymath}
a^2+b^2=c^2
\end{displaymath}
{\settowidth{\parindent}
   {Kde:\ }

\makebox[0pt][r]
 {Kde:\ }$a$, $b$ -- jsou
přilehlé odvěsny pravoúhlého
trojúhelníku.

$c$ -- je přepona pravoúhlého
trojúhelníku.}
```

Je-li potřeba takovéto konstrukce sázet častěji, je vhodné užitím příkazu `\newenvironment` nadefinovat nové prostředí .

3.8 Věty, zákony, ...

Při sazbě matematických textů je pořeba sázet definice, věty, lemma a podobné konstrukce. LaTeX k sazbě takovýchto konstrukcí nabízí příkaz

`\newtheorem{`*jméno*`}[`*čítač*`]{`*text*`}[`*oddíl*`]`

Argument *jméno* je krátké klíčové slovo, které identifikuje danou konstrukci. Argument *text* specifikuje aktuální jméno dané konstrukce, jež se objeví ve výsledném dokumentu.

Argumenty v hranatých závorkách jsou nepovinné. Oba tyto nepovinné parametry nějakým způsobem ovlivňují číslování dané konstrukce. Argumentem *čítač* se lze odvolat na *jméno*, již dříve nadefinované struktury. Argument *oddíl* určuje dělící jednotku, uvnitř které bude daná konstrukce číslována.

Po užití příkazu `\newtheorem` v preambuli dokumentu, lze užít uvnitř vlastního dokumentu následující příkazy.

`\begin{`*jméno*`}[`*text*`]`
Tato důležitá věta matematiky
`\end{`*name*`}`

Skončeme s teorií. Následující příklady vám snad napomohou pochopit k čemu je vlastně dobré prostředí `\newtheorem`.

Zákon 1 *Šéf má vždycky pravdu!*

Přikázání 2 (pro podřízené) *Nemá-li šéf pravdu, viz zákon 1.*

Zákon 3 *Vždy platí zákon 1!*

```
% definitions for the document
% preamble
\newtheorem{zakon}{Zákon}
\newtheorem{prikaz}[zakon]{Přikázání}
%in the document
\begin{zakon} \label{zak:sef}
Šéf má vždycky pravdu!
\end{zakon}
\begin{prikaz}[pro podřízené]
  Nemá-li šéf pravdu,  viz zákon~\ref{zak:sef}.
\end{prikaz}
\begin{zakon}
 Vždy platí zákon~\ref{zak:sef}!
\end{zakon}
```

Konstrukce „Přikázání" užívá stejného čítače jako konstrukce „Zákon", proto „Přikázání" v našem příkladu plynule navazuje na číslování „Zákonů". Další argument v hranatých závorkách určuje další nadpis, či poznámku k nadpisu, dané konstrukce.

Murphy 3.8.1 *Co se může zkazit, to se také zkazí..*

```
\newtheorem{mur}{Murphy}[section]
\begin{mur} Co se může zkazit,
to se také zkazí.. \end{mur}
```

Konstrukce „Murphy" má své číslo odvozené od čísla daného oddílu (section). Samozřejmě, že lze užít i jiné jednotky, např. kapitolu (chapter) nebo pododdíl (subsection).

3.9 Tučné symboly

Získat tučné symboly v LaTeXu je poměrně složité; je to pravděpodobně proto, aby amatérští sazeči nezačali takových možností zneužívat. Příkaz pro změnu tučnosti písma `\mathbf`, avšak toto písmo je vzpřímené, ale matematické symboly se většinou sází matematickou kurzívou (skloněným typem písma). Existuje příkaz `\boldmath`, jež umí pracovat i s matematickými symboly, ale tento příkaz lze užít *pouze vně* matematického režimu.

$$\mu, M \qquad \mathbf{M} \qquad \boldsymbol{\mu, M}$$

```
\begin{displaymath}
\mu, M \qquad \mathbf{M} \qquad
\mbox{\boldmath $\mu, M$}
\end{displaymath}
```

Všimněte si, že čárka je také vysázena tučně, což nemusí být právě to, co jsme očekávali.

Užití balíku amsbsy (součást amsmath) umožňuje prostředky pro pohodlnou práci s tučnými matematickými symboly. Tento balík obsahuje příkaz

`\boldsymbol` a tzv. příkaz poor man's bold `\pmb`, který pracuje aniž by bylo potřeba instalovat nějaké zvláštní fonty pro tučné symboly.

$$\mu, M \qquad \boldsymbol{\mu}, \boldsymbol{M} \qquad \boldsymbol{\mu}, \boldsymbol{M}$$

```
\begin{displaymath}
\mu, M \qquad
\boldsymbol{\mu}, \boldsymbol{M}
\qquad \pmb{\mu}, \pmb{M}
\end{displaymath}
```

3.10 Seznam matematických symbolů

V následujících tabulkách jsou uvedeny všechny symboly, které lze standardně použít v matematickém režimu.

K používání symbolů v tabulkách 3.12–3.16[4], the package je třeba v preambuli dokumentu vložit balík amssymb a samozřejmě je třeba mít nainstalovány matematické AMS fonty. Není-li tento balík v systému nainstalován, lze ho získat na `CTAN:/tex-archive/macros/latex/packages/amslatex`

Tabulka 3.1: Matematické akcenty

$\hat{a}$	`\hat{a}`	$\check{a}$	`\check{a}`	$\tilde{a}$	`\tilde{a}`	$\acute{a}$	`\acute{a}`
$\grave{a}$	`\grave{a}`	$\dot{a}$	`\dot{a}`	$\ddot{a}$	`\ddot{a}`	$\breve{a}$	`\breve{a}`
$\bar{a}$	`\bar{a}`	$\vec{a}$	`\vec{a}`	$\widehat{A}$	`\widehat{A}`	$\widetilde{A}$	`\widetilde{A}`

Tabulka 3.2: Malá řecká písmena

α	`\alpha`	θ	`\theta`	o	`o`	υ	`\upsilon`
β	`\beta`	ϑ	`\vartheta`	π	`\pi`	ϕ	`\phi`
γ	`\gamma`	ι	`\iota`	ϖ	`\varpi`	φ	`\varphi`
δ	`\delta`	κ	`\kappa`	ρ	`\rho`	χ	`\chi`
ϵ	`\epsilon`	λ	`\lambda`	ϱ	`\varrho`	ψ	`\psi`
ε	`\varepsilon`	μ	`\mu`	σ	`\sigma`	ω	`\omega`
ζ	`\zeta`	ν	`\nu`	ς	`\varsigma`		
η	`\eta`	ξ	`\xi`	τ	`\tau`		

Tabulka 3.3: Velká řecká písmena

Γ	`\Gamma`	Λ	`\Lambda`	Σ	`\Sigma`	Ψ	`\Psi`
Δ	`\Delta`	Ξ	`\Xi`	Υ	`\Upsilon`	Ω	`\Omega`
Θ	`\Theta`	Π	`\Pi`	Φ	`\Phi`		

[4]Tyto tabulky byly odvozeny od `symbols.tex` Davida Carlisle, pozměněné tak, jak to doporučil Josef Tkadlec.

Tabulka 3.4: Binární relace

Následující relační operátory mají své negované protějšky. Negovaný operátor se vysází přidáním příkazu `\not` před příslušný symbol.

$<$	`<`	$>$	`>`	$=$	`=`
$\leq$	`\leq` či `\le`	$\geq$	`\geq` či `\ge`	$\equiv$	`\equiv`
$\ll$	`\ll`	$\gg$	`\gg`	$\doteq$	`\doteq`
$\prec$	`\prec`	$\succ$	`\succ`	$\sim$	`\sim`
$\preceq$	`\preceq`	$\succeq$	`\succeq`	$\simeq$	`\simeq`
$\subset$	`\subset`	$\supset$	`\supset`	$\approx$	`\approx`
$\subseteq$	`\subseteq`	$\supseteq$	`\supseteq`	$\cong$	`\cong`
$\sqsubset$	`\sqsubset` [a]	$\sqsupset$	`\sqsupset` [a]	$\Join$	`\Join` [a]
$\sqsubseteq$	`\sqsubseteq`	$\sqsupseteq$	`\sqsupseteq`	$\bowtie$	`\bowtie`
$\in$	`\in`	$\ni$	`\ni` , `\owns`	$\propto$	`\propto`
$\vdash$	`\vdash`	$\dashv$	`\dashv`	$\models$	`\models`
$\mid$	`\mid`	$\parallel$	`\parallel`	$\perp$	`\perp`
$\smile$	`\smile`	$\frown$	`\frown`	$\asymp$	`\asymp`
$:$	`:`	$\notin$	`\notin`	$\neq$	`\neq` či `\ne`

[a] K vysázení těchto symbolů je třeba vložit balík latexsym.

Tabulka 3.5: Binární operátory

$+$	`+`	$-$	`-`		
$\pm$	`\pm`	$\mp$	`\mp`	$\triangleleft$	`\triangleleft`
$\cdot$	`\cdot`	$\div$	`\div`	$\triangleright$	`\triangleright`
$\times$	`\times`	$\setminus$	`\setminus`	$\star$	`\star`
$\cup$	`\cup`	$\cap$	`\cap`	$\ast$	`\ast`
$\sqcup$	`\sqcup`	$\sqcap$	`\sqcap`	$\circ$	`\circ`
$\vee$	`\vee` , `\lor`	$\wedge$	`\wedge` , `\land`	$\bullet$	`\bullet`
$\oplus$	`\oplus`	$\ominus$	`\ominus`	$\diamond$	`\diamond`
$\odot$	`\odot`	$\oslash$	`\oslash`	$\uplus$	`\uplus`
$\otimes$	`\otimes`	$\bigcirc$	`\bigcirc`	$\amalg$	`\amalg`
$\bigtriangleup$	`\bigtriangleup`	$\bigtriangledown$	`\bigtriangledown`	$\dagger$	`\dagger`
$\lhd$	`\lhd` [a]	$\rhd$	`\rhd` [a]	$\ddagger$	`\ddagger`
$\unlhd$	`\unlhd` [a]	$\unrhd$	`\unrhd` [a]	$\wr$	`\wr`

Tabulka 3.6: Velké operátory

$\sum$	`\sum`	$\bigcup$	`\bigcup`	$\bigvee$	`\bigvee`	$\bigoplus$	`\bigoplus`
$\prod$	`\prod`	$\bigcap$	`\bigcap`	$\bigwedge$	`\bigwedge`	$\bigotimes$	`\bigotimes`
$\coprod$	`\coprod`	$\bigsqcup$	`\bigsqcup`			$\bigodot$	`\bigodot`
$\int$	`\int`	$\oint$	`\oint`			$\biguplus$	`\biguplus`

Tabulka 3.7: Šipky

$\leftarrow$	`\leftarrow` či `\gets`	$\longleftarrow$	`\longleftarrow`	$\uparrow$	`\uparrow`
$\rightarrow$	`\rightarrow` či `\to`	$\longrightarrow$	`\longrightarrow`	$\downarrow$	`\downarrow`
$\leftrightarrow$	`\leftrightarrow`	$\longleftrightarrow$	`\longleftrightarrow`	$\updownarrow$	`\updownarrow`
$\Leftarrow$	`\Leftarrow`	$\Longleftarrow$	`\Longleftarrow`	$\Uparrow$	`\Uparrow`
$\Rightarrow$	`\Rightarrow`	$\Longrightarrow$	`\Longrightarrow`	$\Downarrow$	`\Downarrow`
$\Leftrightarrow$	`\Leftrightarrow`	$\Longleftrightarrow$	`\Longleftrightarrow`	$\Updownarrow$	`\Updownarrow`
$\mapsto$	`\mapsto`	$\longmapsto$	`\longmapsto`	$\nearrow$	`\nearrow`
$\hookleftarrow$	`\hookleftarrow`	$\hookrightarrow$	`\hookrightarrow`	$\searrow$	`\searrow`
$\leftharpoonup$	`\leftharpoonup`	$\rightharpoonup$	`\rightharpoonup`	$\swarrow$	`\swarrow`
$\leftharpoondown$	`\leftharpoondown`	$\rightharpoondown$	`\rightharpoondown`	$\nwarrow$	`\nwarrow`
$\rightleftharpoons$	`\rightleftharpoons`	$\iff$	`\iff` (větší mezery)	$\leadsto$	`\leadsto` [a]

[a] K vysázení těchto symbolů je třeba vložit balík latexsym.

Tabulka 3.8: Závorky

$($	`(`	$)$	`)`	$\uparrow$	`\uparrow`	$\Uparrow$	`\Uparrow`
$[$	`[` či `\lbrack`	$]$	`]` či `\rbrack`	$\downarrow$	`\downarrow`	$\Downarrow$	`\Downarrow`
$\{$	`\{` či `\lbrace`	$\}$	`\}` či `\rbrace`	$\updownarrow$	`\updownarrow`	$\Updownarrow$	`\Updownarrow`
$\langle$	`\langle`	$\rangle$	`\rangle`	$\vert$	`|` či `\vert`	$\Vert$	`\\|` či `\Vert`
$\lfloor$	`\lfloor`	$\rfloor$	`\rfloor`	$\lceil$	`\lceil`	$\rceil$	`\rceil`
$/$	`/`	$\backslash$	`\backslash`		. (párová neviditelná)		

Tabulka 3.9: Velké závorky

$\lgroup$	`\lgroup`	$\rgroup$	`\rgroup`	$\lmoustache$	`\lmoustache`	$\rmoustache$	`\rmoustache`
$\arrowvert$	`\arrowvert`	$\Arrowvert$	`\Arrowvert`	$\bracevert$	`\bracevert`		

Tabulka 3.10: Různé symboly

$\dots$	`\dots`	$\cdots$	`\cdots`	$\vdots$	`\vdots`	$\ddots$	`\ddots`
$\hbar$	`\hbar`	$\imath$	`\imath`	$\jmath$	`\jmath`	ℓ	`\ell`
$\Re$	`\Re`	$\Im$	`\Im`	$\aleph$	`\aleph`	$\wp$	`\wp`
$\forall$	`\forall`	$\exists$	`\exists`	$\mho$	`\mho` [a]	∂	`\partial`
$'$	`'`	$\prime$	`\prime`	$\emptyset$	`\emptyset`	∞	`\infty`
∇	`\nabla`	$\triangle$	`\triangle`	$\Box$	`\Box` [a]	$\Diamond$	`\Diamond` [a]
$\bot$	`\bot`	$\top$	`\top`	$\angle$	`\angle`	$\surd$	`\surd`
$\diamondsuit$	`\diamondsuit`	$\heartsuit$	`\heartsuit`	$\clubsuit$	`\clubsuit`	$\spadesuit$	`\spadesuit`
$\neg$	`\neg` or `\lnot`	$\flat$	`\flat`	$\natural$	`\natural`	$\sharp$	`\sharp`

[a] K vysázení těchto symbolů je třeba vložit balík latexsym.

Tabulka 3.11: Nematematické symboly

Tyto symboly lze užít i v textovém režimu.

†	`\dag`	§	`\S`	©	`\copyright`
‡	`\ddag`	¶	`\P`	£	`\pounds`

Tabulka 3.12: AMS — závorky

$\ulcorner$	`\ulcorner`	$\urcorner$	`\urcorner`	$\llcorner$	`\llcorner`	$\lrcorner$	`\lrcorner`

Tabulka 3.13: AMS — řecké a hebrejské znaky

$\digamma$	`\digamma`	$\varkappa$	`\varkappa`	$\beth$	`\beth`	$\daleth$	`\daleth`	$\gimel$	`\gimel`

Tabulka 3.14: AMS — binární relace

$\lessdot$	\lessdot	$\gtrdot$	\gtrdot	$\doteqdot$	\doteqdot či \Doteq
$\leqslant$	\leqslant	$\geqslant$	\geqslant	$\risingdotseq$	\risingdotseq
$\eqslantless$	\eqslantless	$\eqslantgtr$	\eqslantgtr	$\fallingdotseq$	\fallingdotseq
$\leqq$	\leqq	$\geqq$	\geqq	$\eqcirc$	\eqcirc
$\lll$	\lll či \llless	$\ggg$	\ggg či \gggtr	$\circeq$	\circeq
$\lesssim$	\lesssim	$\gtrsim$	\gtrsim	$\triangleq$	\triangleq
$\lessapprox$	\lessapprox	$\gtrapprox$	\gtrapprox	$\bumpeq$	\bumpeq
$\lessgtr$	\lessgtr	$\gtrless$	\gtrless	$\Bumpeq$	\Bumpeq
$\lesseqgtr$	\lesseqgtr	$\gtreqless$	\gtreqless	$\thicksim$	\thicksim
$\lesseqqgtr$	\lesseqqgtr	$\gtreqqless$	\gtreqqless	$\thickapprox$	\thickapprox
$\preccurlyeq$	\preccurlyeq	$\succcurlyeq$	\succcurlyeq	$\approxeq$	\approxeq
$\curlyeqprec$	\curlyeqprec	$\curlyeqsucc$	\curlyeqsucc	$\backsim$	\backsim
$\precsim$	\precsim	$\succsim$	\succsim	$\backsimeq$	\backsimeq
$\precapprox$	\precapprox	$\succapprox$	\succapprox	$\vDash$	\vDash
$\subseteqq$	\subseteqq	$\supseteqq$	\supseteqq	$\Vdash$	\Vdash
$\Subset$	\Subset	$\Supset$	\Supset	$\Vvdash$	\Vvdash
$\sqsubset$	\sqsubset	$\sqsupset$	\sqsupset	$\backepsilon$	\backepsilon
$\therefore$	\therefore	$\because$	\because	$\varpropto$	\varpropto
$\shortmid$	\shortmid	$\shortparallel$	\shortparallel	$\between$	\between
$\smallsmile$	\smallsmile	$\smallfrown$	\smallfrown	$\pitchfork$	\pitchfork
$\vartriangleleft$	\vartriangleleft	$\vartriangleright$	\vartriangleright	$\blacktriangleleft$	\blacktriangleleft
$\trianglelefteq$	\trianglelefteq	$\trianglerighteq$	\trianglerighteq	$\blacktriangleright$	\blacktriangleright

Tabulka 3.15: AMS — šipky

$\dashleftarrow$	\dashleftarrow	$\dashrightarrow$	\dashrightarrow	$\multimap$	\multimap
$\leftleftarrows$	\leftleftarrows	$\rightrightarrows$	\rightrightarrows	$\upuparrows$	\upuparrows
$\leftrightarrows$	\leftrightarrows	$\rightleftarrows$	\rightleftarrows	$\downdownarrows$	\downdownarrows
$\Lleftarrow$	\Lleftarrow	$\Rrightarrow$	\Rrightarrow	$\upharpoonleft$	\upharpoonleft
$\twoheadleftarrow$	\twoheadleftarrow	$\twoheadrightarrow$	\twoheadrightarrow	$\upharpoonright$	\upharpoonright
$\leftarrowtail$	\leftarrowtail	$\rightarrowtail$	\rightarrowtail	$\downharpoonleft$	\downharpoonleft
$\leftrightharpoons$	\leftrightharpoons	$\rightleftharpoons$	\rightleftharpoons	$\downharpoonright$	\downharpoonright
$\Lsh$	\Lsh	$\Rsh$	\Rsh	$\rightsquigarrow$	\rightsquigarrow
$\looparrowleft$	\looparrowleft	$\looparrowright$	\looparrowright	$\leftrightsquigarrow$	\leftrightsquigarrow
$\curvearrowleft$	\curvearrowleft	$\curvearrowright$	\curvearrowright		
$\circlearrowleft$	\circlearrowleft	$\circlearrowright$	\circlearrowright		

Tabulka 3.16: AMS — negované binární relace a šipky

$\nless$	\nless	$\ngtr$	\ngtr	$\varsubsetneqq$	\varsubsetneqq
$\lneq$	\lneq	$\gneq$	\gneq	$\varsupsetneqq$	\varsupsetneqq
$\nleq$	\nleq	$\ngeq$	\ngeq	$\nsubseteqq$	\nsubseteqq
$\nleqslant$	\nleqslant	$\ngeqslant$	\ngeqslant	$\nsupseteqq$	\nsupseteqq
$\lneqq$	\lneqq	$\gneqq$	\gneqq	$\nmid$	\nmid
$\lvertneqq$	\lvertneqq	$\gvertneqq$	\gvertneqq	$\nparallel$	\nparallel
$\nleqq$	\nleqq	$\ngeqq$	\ngeqq	$\nshortmid$	\nshortmid
$\lnsim$	\lnsim	$\gnsim$	\gnsim	$\nshortparallel$	\nshortparallel
$\lnapprox$	\lnapprox	$\gnapprox$	\gnapprox	$\nsim$	\nsim
$\nprec$	\nprec	$\nsucc$	\nsucc	$\ncong$	\ncong
$\npreceq$	\npreceq	$\nsucceq$	\nsucceq	$\nvdash$	\nvdash
$\precneqq$	\precneqq	$\succneqq$	\succneqq	$\nvDash$	\nvDash
$\precnsim$	\precnsim	$\succnsim$	\succnsim	$\nVdash$	\nVdash
$\precnapprox$	\precnapprox	$\succnapprox$	\succnapprox	$\nVDash$	\nVDash
$\subsetneq$	\subsetneq	$\supsetneq$	\supsetneq	$\ntriangleleft$	\ntriangleleft
$\varsubsetneq$	\varsubsetneq	$\varsupsetneq$	\varsupsetneq	$\ntriangleright$	\ntriangleright
$\nsubseteq$	\nsubseteq	$\nsupseteq$	\nsupseteq	$\ntrianglelefteq$	\ntrianglelefteq
$\subsetneqq$	\subsetneqq	$\supsetneqq$	\supsetneqq	$\ntrianglerighteq$	\ntrianglerighteq
$\nleftarrow$	\nleftarrow	$\nrightarrow$	\nrightarrow	$\nleftrightarrow$	\nleftrightarrow
$\nLeftarrow$	\nLeftarrow	$\nRightarrow$	\nRightarrow	$\nLeftrightarrow$	\nLeftrightarrow

Tabulka 3.17: AMS — binární operátory

$\dotplus$	\dotplus	$\centerdot$	\centerdot	$\intercal$	\intercal
$\ltimes$	\ltimes	$\rtimes$	\rtimes	$\divideontimes$	\divideontimes
$\Cup$	\Cup či \doublecup	$\Cap$	\Cap or \doublecap	$\smallsetminus$	\smallsetminus
$\veebar$	\veebar	$\barwedge$	\barwedge	$\doublebarwedge$	\doublebarwedge
$\boxplus$	\boxplus	$\boxminus$	\boxminus	$\circleddash$	\circleddash
$\boxtimes$	\boxtimes	$\boxdot$	\boxdot	$\circledcirc$	\circledcirc
$\leftthreetimes$	\leftthreetimes	$\rightthreetimes$	\rightthreetimes	$\circledast$	\circledast
$\curlyvee$	\curlyvee	$\curlywedge$	\curlywedge		

Tabulka 3.18: AMS — různé symboly

$\hbar$	`\hbar`	$\hslash$	`\hslash`	$\Bbbk$	`\Bbbk`
$\square$	`\square`	$\blacksquare$	`\blacksquare`	$\circledS$	`\circledS`
$\vartriangle$	`\vartriangle`	$\blacktriangle$	`\blacktriangle`	$\complement$	`\complement`
$\triangledown$	`\triangledown`	$\blacktriangledown$	`\blacktriangledown`	$\Game$	`\Game`
$\lozenge$	`\lozenge`	$\blacklozenge$	`\blacklozenge`	$\bigstar$	`\bigstar`
$\angle$	`\angle`	$\measuredangle$	`\measuredangle`	$\sphericalangle$	`\sphericalangle`
$\diagup$	`\diagup`	$\diagdown$	`\diagdown`	$\backprime$	`\backprime`
$\nexists$	`\nexists`	$\Finv$	`\Finv`	$\varnothing$	`\varnothing`
$\eth$	`\eth`	$\mho$	`\mho`		

Tabulka 3.19: Matematická abeceda

Příklad	Příkaz	Potřebný balík
ABCdef	`\mathrm{ABCdef}`	
ABCdef	`\mathit{ABCdef}`	
$\mathnormal{ABCdef}$	`\mathnormal{ABCdef}`	
$\mathcal{ABC}$	`\mathcal{ABC}`	
$\mathscr{ABC}$	`\mathcal{ABC}`	eucal s volbou: mathcal
či	`\mathscr{ABC}`	eucal s volbou : mathscr
$\mathfrak{ABCdef}$	`\mathfrak{ABCdef}`	eufrak, amsfonts či amssymb
$\mathbb{ABC}$	`\mathbb{ABC}`	amsfonts či amssymb

Kapitola 4

Speciality

Nyní byste měli být vybaveni dostatkem informací k vlastnímu napsání článku, knihy, referátu v LaTeXu. Tato kapitola je zde jaksi navíc a proto není nutné ji číst. Teprve až při psaní svých dokumentů narazíte na nějaký problém, který nelze vyřešit dosud zmíněnými prostředky, nahlédněte do této kapitoly. Avšak ani tato kapitola neobsahuje zdaleka všechny jemnosti LaTeXu. Kompletnější informace naleznete v knihách *LaTeX Manual* [1] a *The LaTeX Companion* [3].

4.1 Druhy a velikosti písma

LaTeX volí vhodný druh a velikost písma na základě příkazů, které udávají logickou strukturu textu (nadpisy, zdůraznění atd.). Ve zvláštních případech lze změnit druh, nebo velikost písma, pomocí příkazů uvedených v tab. 4.1 a 4.2. Druh písma, který LaTeX v daném dokumentu užije, zavisí do značné míry také na užité třídě, všech různých volbách třídy a na dalších užitých balíčcích.

Malí **tuční** Římané ovládli celou velkou *Itálii.*

```
{\small Malí
\textbf{tuční} Římané ovládli}
{\Large celou velkou
\textit{Itálii}.}
```

Jedním důležitým rysem LaTeXu je, že atributy písma jsou nezávislé. To znamená, že lze změnit např. velikost písma, přičemž se zachová atribut tloušťky či sklonu písma nastavené dříve. Tato věc se moždá zdá zřejmá lidem, jež se učí LaTeX 2_ε od začátku, avšak nemusí to být úplně jasné lidem přecházející na LaTeX 2_ε z LaTeXu 2.09.

Užitím příkazů pro změnu typu písma v *matematickém režimu* lze dočasně tento mód opustit a tak vysázet do matematiky běžný text. Je-li třeba přejít na jiný typ písma uvnitř matematického režimu, je třeba užít zvláštních příkazů; viz tab. 4.3.

Tabulka 4.1: Druhy písma

`\textrm{...}`	normální písmo — antikva (roman)
`\textsf{...}`	bezserifové písmo (sans serif)
`\texttt{...}`	`strojopisné písmo (typewriter)`
`\textmd{...}`	střední písmo (medium)
`\textbf{...}`	**polotučné písmo (bold face)**
`\textup{...}`	vzpřímené písmo (upright)
`\textit{...}`	*kurzíva (italic)*
`\textsl{...}`	*skloněné písmo (slanted)*
`\textsc{...}`	KAPITÁLKY (SMALL CAPS)
`\emph{...}`	*zvýrazněné písmo (emphasised)*
`\textnormal{...}`	běžné písmo dokumentu font

Tabulka 4.2: Velikosti písma

`\tiny`	nejmenší písmo
`\scriptsize`	velmi malé písmo (velikost indexů)
`\footnotesize`	malé písmo (velikost pozmámek pod čarou)
`\small`	menší písmo
`\normalsize`	písmo běžné velikosti
`\large`	velké písmo
`\Large`	větší písmo
`\LARGE`	velké písmo
`\huge`	velmi velké písmo
`\Huge`	největší písmo

Ve spojení s těmito příkazy pro změnu typu písma, hrají nemalou roli složené závorky. Ty se užívají k vytváření tzv. *skupin.* Skupiny ohraničují platnost většiny příkazů LaTeXu.

Má rád velká i malá písmenka.

```
Má rád {\LARGE velká i
{\small malá} písmenka}.
```

Se změnou velikosti písma se mění také řádkování, ale jen v případě, že odstavec končí ještě uvnitř platnosti příkazu pro změnu velikosti písma. Proto není dobré uzavírací složenou závorku `}` vkládat do vstupního textu příliš brzy. Všimněte si umístění příkazu `\par` v následujících dvou příkladech.

Nečtěte to! Není to pravda! To mi můžete věřit!

```
{\Large Nečtěte to! Není to pravda!
To mi můžete věřit!\par}
```

Toto není pravda. Avšak nezapomeňte, že jsem lhář!

```
{\Large Toto není pravda. Avšak
  nezapomeňte, že jsem lhář!}\par
```

Závěrem naší exkurze do světa písma přijměte tuto radu[1]:

> **Zapamatujte si!** *Čím* VÍCE druhů písma použijete **ve** svém `textu,` *tím* se stane ČITELNĚJŠÍM a *hezčím.*

[1]Pozor: Toto je myšleno s nadsázkou, jak jistě bystrý čtenář ihned odhalil.

Tabulka 4.3: Matematická písma

Příkaz	*Příklad*	*Výstup*
`\mathcal{...}`	`$\mathcal{B}=c$`	$\mathcal{B}=c$
`\mathrm{...}`	`$\mathrm{K}_2$`	K_2
`\mathbf{...}`	`$\sum x=\mathbf{v}$`	$\sum x=\mathbf{v}$
`\mathsf{...}`	`$\mathsf{G\times R}$`	$\mathsf{G\times R}$
`\mathtt{...}`	`$\mathtt{L}(b,c)$`	$\mathtt{L}(b,c)$
`\mathnormal{...}`	`$\mathnormal{R_{19}}\neq R_{19}$`	$\mathnormal{R_{19}}\neq R_{19}$
`\mathit{...}`	`$\mathit{ffi}\neq ffi$`	$\mathit{ffi}\neq ffi$

4.2 Mezery

4.2.1 Mezery mezi řádky

Je-li třeba docílit větších mezer mezi sázenými řádky, je vhodné užít příkazu

```
\linespread{faktor}
```

v preambuli dokumentu. Pro řádkování „jedna a půl" se užije `\linespread{1.3}` a pro řádkování „dva" `\linespread{1.6}`. Běžně však není nutné řádky od sebe oddalovat, proto je standardně *faktor* oddálení řádek nastaven na hodnotu 1.

4.2.2 Formátování odstavce

V LaTeXu jsou dva parametry, které ovlivňují vzhled odstavce. Umístěním definice, která může vypadat například takto:

```
\setlength{\parindent}{0pt}
\setlength{\parskip}{1ex plus 0.5ex minus 0.2ex}
```

v preambuli dokumentu[2] lze změnit vzhled všech odstavců. V těchto dvou definicích jsme nejprve zakázali odsazení začátku každého odstavce (nastavili jsme hodnotu registru `\parindent` na nulu) a poté jsme zvětšili mezeru mezi dvěma odstavci (nastavili jsme hodnotu registru `\parskip` na 1ex s mírnou vůlí. Takovýto formát odstavců — neodsazované odstavce oddělené mezerou — je v Evropě běžný. Avšak pozor, toto nastavení má vliv i na sazbu tabulky obsahu, kde vlastně každá řádka obsahu je odstavcem. Proto se řádky v obsahu od sebe také vzdálí o hodnotu nastavenou v `\parskip`.

Je-li třeba odsadit odstavec, který není automaticky odsazen, je dobré užít na začátku takového odstavce příkaz:

```
\indent
```

Tento příkaz lze užít v případě, že `\parindent` je nenulové. Například, chceme-li, aby na počátku dané sekce byl první odstavec také odsazen.[3]

Chceme-li v daném odstavci zakázat jeho odsazení, uvedeme jako první příkaz tohoto odstavce:

```
\noindent
```

Toho se užívá v případě, že se dokument začíná čistým textem a nikoli nějakým příkazem pro strukturalizaci dokumentu.

[2] Mezi příkazy `\documentclass` a `\begin{document}`.

[3] Chceme-li dosáhnout toho, aby byl odsazen první odstavec každé sekce, je lépe použít balíku indentfirst ze souboru balíků „tools".

4.2.3 Horizontální mezery

LaTeX sám nastavuje velikost mezer mezi slovy a větami. Někdy je však potřeba přidat do textu horizontální mezeru, k tomu se užívá příkaz:

`\hspace{`*délka*`}`

Je-li třeba mezeru zachovat i v případě, že připadne na konec či začátek řádky, užije se místo příkazu `\hspace` jeho ohvězdičkované verze tj. `\hspace*`. Parametr *délka* je v nejjednodužším případě číslice následovaná jednotkou délky. Nejdůležitější jednotky jsou uvedeny v tabulce 4.4.

Zde je   mezera 1.5 cm.

```
Zde je\hspace{1.5cm}mezera 1.5~cm.
```

Příkaz

`\stretch{`*n*`}`

umožňuje vytvořit zvláštní druh mezery — pružnou mezeru. Tato mezera má schopnost roztáhnout se tak, že zaplní veškeré zbylé místo na daném řádku. LaTeX zavádí pro `\stretch{1}` ekvivalent `\fill` a pro `\hspace{\fill}` ekvivalent `\hfill`.

A B   CD
A B   CD
A B   CD
A B   CD

```
A B \hspace{\stretch{3}} CD\\
A B \hspace{\stretch{1}} CD\\
A B \hspace{\fill} CD\\
A B \hfill{} CD\\
```

Jestliže se v tom samém řádku sejde více příkazů typu `\hspace{\stretch{`*n*`}}`, příslušné pružné mezery se napnou v poměru příslušných faktorů *n*.

A   B   C   D

```
A\hspace{\stretch{2}}
B\hspace{\stretch{1}}
C\hspace{\stretch{3}}D
```

Pro sazbu některých seznamů, například obsahu, je třeba pružně vyplnit místo na řádku tečkami či linkou, přižemž předem není známo kolik bude potřeba teček či jak dlouhou linku bude třeba užít. Pro tento účel LaTeX zavádí příkazy `\dotfill` a `\hrulefill`.

A ______ B   C . . . . . . D

```
A \hrulefill{} B
\hspace{\stretch{2}}
C \dotfill{}   D
```

Tabulka 4.4: Jednotky délky užívané v TeXu

`mm`	milimetr = 1/25,4 in
`cm`	centimetr = 10 mm
`in`	palec (inch, coul) = 25,4 mm
`pt`	monotypový bod (point) = 1/72,27 in $\approx \frac{1}{3}$ mm
`em`	$\approx$ šířka písmene `M` aktuálního typu písma
`ex`	$\approx$ výška písmene `x` aktuálního typu písma

4.2.4 Vertikální mezery

Mezery mezi odstavci, kapitolami atd. určuje LaTeX sám. Ve zvláštních případech lze zvětšit mezeru *mezi dvěma odstavci* příkazem:

`\vspace{`*délka*`}`

Tento příkaz bychom měli zadávat vždy mezi dvěma prázdnými řádky. Pokud má mezera zůstat také na začátku nebo na konci stránky, musíme namísto `\vspace` použít jeho ohvězdičkovanou verzi, tj. `\vspace*`.

Příkazy v tab. 4.5 vytvářejí speciální vertikální mezery.

Tabulka 4.5: Zvláštní vertikální mezery

`\smallskip`	přibližně 1/4 řádku
`\medskip`	přibližně 1/2 řádku
`\bigskip`	přibližně 1 řádek
`\vfill`	ekvivalent k `\vspace{\stretch{1}}`

Příkaz `\stretch` ve spojení s `\pagebreak` slouží k vysázení textu na dolní okraj stránky nebo k vertikálnímu centrování textu.

```
Nějaký text \ldots

\vspace{\stretch{1}}
Tento text bude vysázen na poslední řádku stránky.\pagebreak
```

Mezeru mezi řádky *uvnitř* odstavce či tabulky můžeme zvětšit příkazem:

`\\[`*délka*`]`

4.3 Vzhled stránky

LaTeX 2_ε umožňuje nastavit rozměr papíru, na nejž chceme text sázet, přímo v příkazu `\documentclass`. Podle této specifikace si pak sám nastaví okraje. Avšak někdy autor potřebuje tyto přednastavené hodnoty upravit podle svých představ. Na obrázku 4.1 jsou zobrazeny veškeré nastavitelné parametry, které ovlivňují vzhled stránky. Tento obrázek byl vytvořen s využitím balíku layout, který je součástí souboru balíků tools[4].

LaTeX nabízí dva příkazy pomocí nichž lze tyto parametry měnit. Obyčejně se objevují v preambuli dokumentu.

První příkaz umožňuje přiřadit libovolnému z těchto parametrů pevnou hodnotu:

`\setlength{`*parameter*`}{`*délka*`}`

Druhý umožňuje zvětšit libovolný z těchto parametrů o pevný rozměr *délka*.

`\addtolength{`*parameter*`}{`*délka*`}`

Tento příkaz je možná užitečnější než `\setlength`, neboť umožňuje nastavovat rozměry vzhledem k již nastaveným hodnotám. Přidání jednoho centimetru k celkové šířce textu by se provedlo vložením těchto příkazů do preambule dokumentu:

```
\addtolength{\hoffset}{-0.5cm}
\addtolength{\textwidth}{1cm}
```

4.4 Sazba seznamu použité literatury

Seznam použité literatury se vysází pomocí prostředí `thebibliography`. Každá položka v seznamu literatury začíná příkazem:

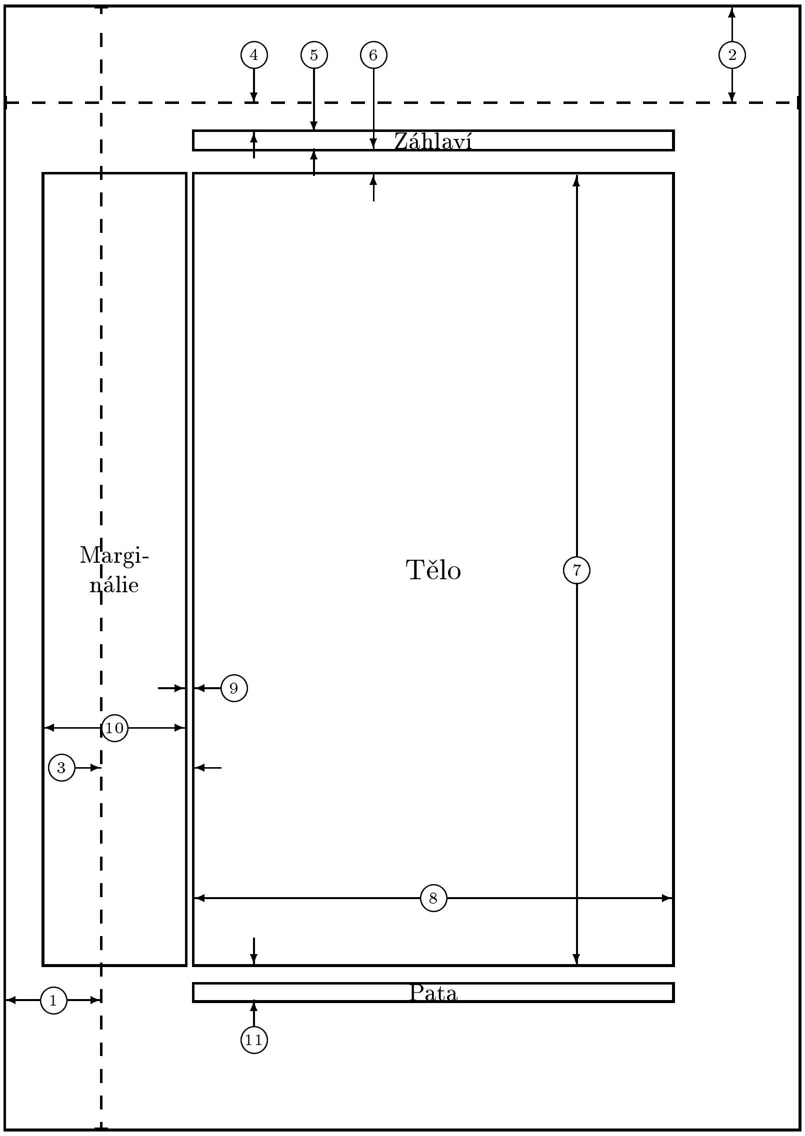
`\bibitem{`*značka*`}`

kde *značka* se používá v těle dokumentu při odkazu na uvedené dílo.

`\cite{`*značka*`}`

Číslování položek seznamu se děje automaticky, přičemž parametrem prostředí `thebibliography` se nastaví maximální šířka tohoto čísla.

[4] `CTAN:/tex-archive/macros/latex/packages/tools`



1	`jeden palec + \hoffset`	2	`jeden palec + \voffset`
3	`\evensidemargin = 70pt`	4	`\topmargin = 22pt`
5	`\headheight = 13pt`	6	`\headsep = 19pt`
7	`\textheight = 595pt`	8	`\textwidth = 360pt`
9	`\marginparsep = 7pt`	10	`\marginparwidth = 106pt`
11	`\footskip = 27pt`		`\marginparpush = 5pt (nevyobrazen)`
	`\hoffset = 0pt`		`\voffset = 0pt`
	`\paperwidth = 597pt`		`\paperheight = 845pt`

Obrázek 4.1: Parametry ovlivňující vzhled stránky

Partl [1] dodává, že ...

```
Partl~\cite{pa}
dodává, že  \ldots

\begin{thebibliography}{99}
\bibitem{pa} H.~Partl:
\emph{German \TeX},
TUGboat Vol.~9, No.~1 ('88)
\end{thebibliography}
```

Literatura

[1] H. Partl: *German TEX*, TUGboat Vol. 9, No. 1 ('88)

4.5 Tvorba rejstříku

Velmi užitečnou součástní většiny knih je rejstřík. Pomocí LATEXu a podpůrného programu `makeindex`,[5] lze vytvořit rejstřík (index) poměrně jednoduše. Tento úvod objasňuje pouze základy tvorby indexů, daleko podrobnější popis naleznete např. v *The LATEX Companion* [3].

Aby bylo možno začít užívat automatickou tvorbu rejstříku v LATEXu je třeba v preambuli dokumentu užít balík makeidx, tj. vložit příkaz:

```
\usepackage{makeidx}
```

Dále je třeba v preambuli vložit příkaz,

```
\makeindex
```

který spustí tvorbu příslušných souborů pro tvorbu indexu.

Obsah rejstříku se tvoří pomocí příkazu:

`\index{`*položka*`}`

Kde argument *položka* je jedna položka, kterou chceme vložit do rejstříku. Tento příkaz se vkládá do toho místa v textu, na které chceme nasměrovat odkaz v rejstříku. Tabulka 4.6 na několika příkladech vysvětluje syntax argumentu *položka*.

Během zpracování vstupního souboru LATEXem každý příkaz `\index` zapisuje příslušný zápis názvu položky spolu s číslem aktuální stránky do

[5] V systémech, kreré neumožňují mít jména souborů delší jak osm znaků se program jmenuje `makeidx`.

Tabulka 4.6: Příklady položek vkládaných do rejstříku

Příklad	**Položka v rejstříku**	**Poznámka**
`\index{jméno}`	jméno, 1	jednoduchá položka
`\index{jméno!Petr}`	Petr, 3	subpoložka položky „jméno“
`\index{Sam@\textsl{Sam}}`	*Sam*, 2	formátovaná položka
`\index{Jan@\textbf{Jan}}`	**Jan**, 7	
`\index{Jana|textbf}`	Jena, **3**	formátované číslo stránky
`\index{Jiří|textit}`	Jiří, *5*	

zvláštního souboru. Tento soubor má stejné jméno jako hlavní vstupní soubor a příponu `.ind`. Tento soubor je poté třeba zpracovat programem `makeindex`:

`makeindex` *jméno_souboru*

Program `makeindex` vytvoří uspořádaný rejstřík a uloží jej do souboru s příponou `.idx`. Je-li nyní příslušný vstupní soubor znovu zpracován LaTeXem, je tento uspořádaný resjtřík vysázen v místě výskytu příkazu:

```
\printindex
```

Užijeme-li balík showidx, pak LaTeX vytiskne indexovou položku navíc na levý okraj aktuálně sázené stránky. Tato vlastnost je velice užitečná při tvorbě rejstříku a korektůr.

4.6 Vkládání obrázků ve formátu EPS

Prostředí `figure` a `table` LaTeXu nabízí základní prostředky pro práci s plovoucími objekty (obrázky, grafika).

V současné době existuje velké množství možností jakými lze vytvářet grafický výstup (jednoduché obrázky, grafy,...) přímo pomocí příkazů LaTeXu či některého z jeho podpůrných balíků. Žel většině začínajících uživatelů LaTeXu se tento způsob zdá poněkud složitý a nepochopitelný — většinou totiž jde o příkazy umožňující vektorový popis daného obrázku. Proto se v tomto manuálku nebudeme tímto způsobem tvorby grafiky zabývat, pouze zvídavého čtenáře odkážeme na vynikající knihy, *The LaTeX Companion* [3] a *LaTeX Manual* [1], kde lze nalézt podrobnosti.

Asi nejjednožším způsobem, jak dostat grafiku do dokumentu, je vytvořit obrázek specializovaným programem[6] a výsledek poté vložit do dokumentu. I v tomto případě nejste v LaTeXu omezeni jen na jeden způsob takového vložení předem vytvořeného obrázku. V tomto úvodu se však omezíme jen na vkládání grafiky ve formě encapsulated PostScriptu (EPS) , jelikož tento způsob je asi nejužívanějším a možná i nejjednodužším.

Užijeme-li tohoto přístupu, je třeba po zpracování LaTeXem převést výstup (soubor `.dvi`) do formátu PostScriptu a poté použít postscriptovou tiskárnu nebo programu GhostScript[7]. který umožňuje tisk PostScriptu téměř na libovolném typu tiskárny.

Užitečnou sadu příkazů pro vkládání grafiky poskytuje balík graphicx D. P. Carlislea. Tento balík je součástí celé rodiny balíků, která se označuje jako „graphics bundle“[8].

V dalším předpokládejme, že pracujeme na systému, který má k dispozici PostSriptovou tiskárnu, je naistalován balík graphicx. Pak lze k vkládání grafiky do dokumentu užít následujících kroků:

1. Vyexportujeme obrázek z grafického programu ve formátu EPS.

2. Vložíme balík maker graphicx, tj. do preambule zdrojového textu vložíme příkaz

   `\usepackage[`*driver*`]{graphicx}`

   driver je jméno programu, který bude užit ke koverzi dvi-souboru do postscriptu. [9]. Tuto informaci musíme balíku zdělit proto, že vlastní vkládání grafiky do tištěného dokumentu zajišťuje právě ovladač tisku, tj. konvertor DVI na příkazy tiskárny, v našem případě na příkazy postscriptu. Tím, že je znám ovladač *driver*, je zajištěno, že balík graphicx vloží do dvi-souboru správné příkazy na vložení požadovaného EPS obrázku.

Tabulka 4.7: Názvy klíčů pro balík graphicx

`width`	úprava šířky vkládané grafiky na specifikovanou hodnotu
`height`	úprava výšky vkládané grafiky na specifikovanou hodnotu
`angle`	rotace obrázku ve směru hodinových ručiček

[6] Např. XFig, CorelDraw!, Freehand, Gnuplot, ...
[7] `CTAN:/tex-archive/support/ghostscript`
[8] `CTAN:/tex-archive/macros/latex/packages/graphics`
[9] Nejužívanějším programem užívaným ke konverzi DVI na PS je program `dvips`.

3. Pro vložení souboru *file* s grafikou ve formátu EPS užijeme příkaz:

   `\includegraphics[`*klíč*=*hodnota*, ...`]{`*file*`}`

   Volitelným parametrem je seznam čárkou oddělených dvojic *klíč*=*hodnota*. Pomocí těchto klíčů lze měnit šířku, výšku a rotaci vkládané grafiky. V tabulce 4.7 je uveden seznam nejdůležitějších klíčů.

Pro názornost uveďme příklad:

```
\begin{figure}
\begin{center}
\includegraphics[angle=90, width=10cm]{test.eps}
\end{center}
\end{figure}
```

Tímto kódem jsme do textu vložili grafiku uloženou v souboru `test.eps`. Obrázek je nejprve otočen o devadesát stupňů po směru hodinových ručiček a *poté* je upravena jeho velikost na šířku 10 cm, protože není explicitè uvedena výška, změní v příslušném poměru.

Více informací lze nalézt v [9], [3] a [4].

Literatura

[1] Leslie Lamport. *LaTeX: A Document Preparation System.* Addison-Wesley, Reading, Massachusetts, second edition, 1994, ISBN 0-201-52983-1.

[2] Donald E. Knuth. *The TeXbook,* Volume A of *Computers and Typesetting*, Addison-Wesley Publishing Company (1984), ISBN 0-201-13448-9.

[3] Michel Goossens, Frank Mittelbach and Alexander Samarin. *The LaTeX Companion.* Addison-Wesley, Reading, Massachusetts, 1994, ISBN 0-201-54199-8.

[4] Michel Goossens, Frank Mittelbach. *The LaTeX Graphic Companion.* Addison-Wesley, Reading, Massachusetts, ISBN 0-201-85469-4,.

[5] Ke každé instalaci by měl být přibalen manuál *LaTeX Local Guide*, ve kterém jsou uvedeny údaje specifické pro danou instalaci — např. příkazy potřebné pro spuštění programů a soubory, předdefinované formáty a fonty, které jsou k dispozici. Měl by být uložen v souboru `local.tex`. Žel, většina systémových administrátorů takový dokument nepořizuje. V takovém je nutné kontaktovat nějakého LaTeXového guru, většinou jím bývá právě systémový administrátor.

[6] LaTeX3 Project Team. *LaTeX 2ε for authors.* Je součástí distribuce LaTeX 2ε jako soubor `usrguide.tex`.

[7] LaTeX3 Project Team. *LaTeX 2ε for Class and Package writers.* Je součástí distribuce LaTeX 2ε jako soubor `clsguide.tex`.

[8] LaTeX3 Project Team. *LaTeX 2ε Font selection.* Je součástí distribuce LaTeX 2ε jako soubor `fntguide.tex`.

[9] D. P. Carlisle. *Packages in the 'graphics' bundle.* Je součástí distribuce souboru maker ‚graphics' jako soubor `grfguide.tex`.

[10] H. Partl, E.Schlegl, I. Hyna, P. Sýkora, *LaTeX Stručný popis,* Je součástí distribuce CSTeXu jako soubor `uvodlat.zip`

[11] Jiří Rybička, *LaTeX pro začátečníky,* 2. vydání Konvoj 1999, ISBN 80-85615-77-0 (brož.), ISBN 80-85615-74-6 (váz.)

[12] Petr Olšák, *Typografický systém TeX,* CsTUG 1995, ISBN 80-901950-0-8

[13] Petr Olšák, *TeXbook naruby,* Konvoj 1997, ISBN 80-85615-64-9

Rejstřík