UNIVERZITA PALACKÉHO V OLOMOUCI
PŘÍRODOVĚDECKÁ FAKULTA

BAKALÁŘSKÁ PRÁCE

Sazba matematických formulí plainTEXem

Katedra matematické analýzy a aplikací matematiky
Vedoucí bakalářské práce: **RNDr. Miloslav Závodný**
Vypracoval(a): **Aneta Václavková**
Studijní program: B1103 Aplikovaná matematika
Studijní obor Matematika–ekonomie se zaměřením na bankovnictví
Forma studia: prezenční
Rok odevzdání: 2015

BIBLIOGRAFICKÁ IDENTIFIKACE

Autor: Aneta Václavková

Název práce: Sazba matematických formulí plainTEXem

Typ práce: BAKALÁŘSKÁ PRÁCE

Pracoviště: Katedra matematické analýzy a aplikací matematiky

Vedoucí práce: RNDr. Miloslav Závodný

Rok obhajoby práce: 2015

Abstrakt: PlainTEX je Knuthův formát (systém maker) pro sazbu textů obsahujících sazbu matematickou. Pomocí něho lze pohodlně sázet matematické formule. Uživatel, více obeznámený s tímto formátem, může jeho makra výhodně užít i v nejrozšířenějším formátu LATEX. Obsahem této práce je trochu hlubší vysvětlení činnosti PlainTEXu při sazbě matematiky. Je tedy určena čtenáři poučenému v práci s TEXem.

Klíčová slova: PlainTEX, matematická sazba, formule

Počet stran: 41

Počet příloh: 0

Jazyk: český

BIBLIOGRAPHICAL IDENTIFICATION

Author: Aneta Václavková

Title: Typing math formulas by plainTEX

Type of thesis: Bachelor's

Department: Department of Mathematical Analysis and Application of Mathematics

Supervisor: RNDr. Miloslav Závodný

The year of presentation: 2015

Abstract: PlainTEX is the Knuth's format (macro system) for typesetting of texts which contain the mathematical rate. You can use it for comfortably typesetting of mathematical formulas. The user, who is more acquainted with this format, can use advantageously the macros in the most widely used LaTEX format. The main content of this bachelor thesis is the deeper explan ation of plainTEX for typesetting of mathematics rate. It´s been intended for an informed reader who wants to work with TEX.

Key words: PlainTEX, displaying math, math formulas

Number of pages: 41

Number of appendices: 0

Language: Czech

Prohlášení

Prohlašuji, že jsem bakalářskou práci zpracovala samostatně pod vedením pana RNDr. Miloslava Závodného a všechny použité zdroje jsem uvedla v seznamu literatury.

V Olomouci dne ....

podpis

Obsah

Úvod **7**

1 Matematická sazba **8**
1.1 Atomy . . . 9
1.1.1 Běžné matematické značky – třída 0 . . . 10
1.1.2 Velké matematické operátory – třída 1 . . . 11
1.1.3 Binární operátory – třída 2 . . . 11
1.1.4 Relace – třída 3 . . . 12
1.1.5 Otevírací a zavírací závorky – třída 4 a 5 . . . 13
1.2 Písmo v matematickém režimu . . . 14

2 Sazba matematických formulí **17**
2.1 Horní a dolní indexy . . . 17
2.2 Sazba s akcentem . . . 18
2.3 Matematické styly . . . 18
2.4 Sazba matematických objektů . . . 20
2.4.1 Mezery v matematické sazbě . . . 20
2.4.2 Sazba názvů matematických funkcí . . . 20
2.4.3 Zlomky . . . 21
2.4.4 Řetězové zlomky . . . 21
2.4.5 Odmocniny a mocniny . . . 22
2.4.6 Horizontální oddělovače . . . 23
2.4.7 Vertikální oddělovače . . . 24
2.4.8 Symboly nad a pod objektem . . . 27
2.4.9 Další relace . . . 27
2.4.10 Limity . . . 28
2.5 Fantóm – prázdný box . . . 30
2.5.1 Výpustky . . . 30

3 Sazba víceřádkových formulí **31**
3.1 Zarovnání víceřádkových formulí . . . 33
3.2 Značení vzorců – číslování rovnic . . . 34
3.3 Matice . . . 36

Závěr **40**

Literatura **41**

Poděkování

Ráda bych poděkovala vedoucímu bakalářské práce panu RNDr. Miloslavu Závodnému za spolupráci i za čas, který mi věnoval při konzultacích.

Úvod

Tématem mé bakalářské práce je *Sazba matematických formulí plainTEXem*. Toto téma jsem si vybrala, protože se mi zalíbily možnosti, které tento program nabízí pro sazbu matematiky.

Cílem bakalářské práce bylo přehledným a logickým způsobem zpracovat možnosti, které poskytuje plainTEX pro sazbu matematických formulí, a tyto možnosti demonstrovat na příkladech.

Práci jsem rozdělila do 3 kapitol. V první kapitole jsem popsala základní elementy matematické sazby – atomy, jejich strukturu, písmo používané v matematickém režimu. Ve druhé kapitole jsem se zabývala sazbou jednořádkových formulí, vše jsem ilustrovala na příkladech. Ve třetí kapitole jsem se zaměřila na sazbu víceřádkových formulí.

Nakonec jsem uvedla literaturu, která byla využita při psaní této práce. Může sloužit jako užitečný zdroj informací i v oblastech nesouvisejících přímo se sazbou formulí.

1. Matematická sazba

Kromě vertikálního a horizontálního módu má TEX k dispozici i mód matematický, určený pro sazbu matematických formulí. Chování TEXu v tomto módu je značně odlišné od jeho práce ve vertikálním a horizontálním módu.

Do matematického módu TEX přejde po přečtení znaku $ (matematický textový mód), kde jsou vzorce sázeny v rámci odstavce, nebo dvojice dolarů $$ (matematický vysazený mód), kde jsou vzorce sázeny na samostatný řádek oddělený mezerou od okolního textu. V matematickém módu sestavuje TEX tzv. matematický seznam [4].

Matematický seznam může obsahovat tyto elementy [3]:

- Atom – základní nositel strukturované informace.
- Značky typu `\left`, `\right`, `\scriptstyle` atd.
- Prvky z horizontálního seznamu (linky, boxy, penalty, značky pro dělení slov).
- Pružné výplňky (`\hskip`, `\hfil`, `\dots`, `\mskip`, aj.).
- Pevné výplňky (`\kern`, `\mkern`).
- Značky jako v horizontálním či vertikálním seznamu. (`\write`, `\insert`, aj.).

Základním elementem matematické sazby je *atom*. Každý atom sestává ze tří částí:

- Základ (nucleus) – objekt, ke kterému jsou připojovány indexy a exponenty.
- Exponent (superscript).
- Index (subscript).

Každá z těchto tří částí může obsahovat definitivní sazbu jednoho matematického znaku nebo může obsahovat matematický seznam. Do každé části lze vkládat další atomy a sestavovat tak libovolné formule, část atomu muže být i prázdná. Úroveň vnoření není omezena.

Například: `$y^2_{k+2}$` – atom má základ y, exponent je 2 a index je matematický seznam s atomy k, +, 2; `$y^2$` – má základ y, exponent 2 a index je prázdný; `${}^2_{a}$` – základ je prázdný, exponent 2 a index a.

1.1. Atomy

TEX rozděluje atomy a znaky do 13 tříd [3]:

0. Ord – běžné matematické značky (název proměnné x, a, aj.).
1. Op – velké matematické operátory (např. $\sum$, $\prod$, $\int$).
2. Bin – binární operace (např. $+$, $-$, $:$, $\times$).
3. Rel – relace, kolem je z obou stran vetší mezera (např. $=$, $<$, $\geq$).
4. Open – otevírací závorky ($\langle$, $\{$).
5. Close – zavírací závorky ($\rangle$, $\}$).
6. Punct – interpunkce (čárka, středník, :).
7. Inner – výsledek sazby se závorkami proměnlivé výšky `\left`, `\right`.
8. Over – atom bude sázen s čarou nad ($\overline{a+b}$).
9. Under – atom bude sázen s čarou pod ($\underline{a-b}$).
10. Acc – atom bude opatřen specifikovaným akcentem ($\widetilde{xy}+\hat{a}$).
11. Rad – atom bude sázen jako odmocnina ($\sqrt{x}$).
12. Vcent – sazba do boxu centrovaná na matematickou osu (`\vcenter{...}`).

Atomy typu 0–6 jsou vytvářeny automaticky při činnosti povelu pro sazbu znaku. Pro další typy atomů jsou v TEXu rezervovány speciální primitivy.

Definujeme-li vlastní značky pro sazbu symbolu v matematickém režimu, musíme mít na zřeteli, že symbol musí být korektně vysázen, je tedy třeba mu přiřadit požadovanou třídu.

Relaci . definujeme `\def\ldot{\mathrel{.}}`, pak `$a\ldot b$` vysází $a \ldot b$.

Operátor tg definujeme `\def\tg{\mathop{\hbox{\rm tg}}}`, potom `$\tg x$` vysází $\tg x$.

Otevírací a zavírací znaky definujeme např.:

```
\newcommand{\dlangl}{\mathopen{<}}
\newcommand{\drangl}{\mathclose{>}}.
```

Potom `$\dlangl a,b\drangl$` vysází $<a, b>$. Napíšeme-li `$<a,b>$`, dostaneme $< a, b >$; vidíme, že znaky < a > jsou v plainu definovány jako relace. Každý symbol má pro použití v matematickém režimu třídu přiřazenou. Předdefinovanou třídu znaku lze potlačit jeho uvedením ve složených závorkách, pak je z něj běžný znak. Srovnejte `$a{:}b$` a `$a:b$`, tj. $a{:}b$ a $a : b$. Píšeme-li číslo s desetinnou čárkou, musíme zadat např. `$1{,}2$`, aby se vysázelo správně 1,2, čárka je oddělovač (TEX předpokládá desetinnou tečku).

TEX totiž při sazbě matematiky dává kolem objektů, které mají v matematické sazbě konkrétní význam, takové mezery, aby sazba odpovídala sazbě historicky „nejkrásnějších“ matematických knih [3]:

1. Kolem relací má být z obou stran větší mezera.
2. Kolem binárních operací má být z obou stran střední mezera.
3. Za čárkou má být malá mezera.
4. Kolem velkých operátorů má být malá mezera.
5. Kolem vzorců se závorkami `\left`, `\right` mají být malé mezery.

Všimněme si nyní povelů pro sazbu znaků prvních 5 tříd.

1.1.1. Běžné matematické značky – třída 0

1. Běžné znaky zahrnují písmena řecké abecedy [1]:

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

Mezi velkými písmeny řecké abecedy je 11 písmen, které nevychází z latinky [1]:

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

2. Další matematické symboly [1]:

$\aleph$	\aleph	$\prime$	\prime	$\forall$	\forall
$\hbar$	\hbar	$\emptyset$	\emptyset	$\exists$	\exists
$\imath$	\imath	∇	\nabla	$\neg$	\not nebo \lnot
$\jmath$	\jmath	$\surd$	\surd	$\flat$	\flat
ℓ	\ell	$\top$	\top	$\natural$	\natural
$\wp$	\wp	$\bot$	\bot	$\sharp$	\sharp
$\Re$	\Re	$\Vert$	\Vert	$\clubsuit$	\clubsuit
$\Im$	\Im	$\angle$	\angle	$\diamondsuit$	\diamondsuit
∂	\partial	$\triangle$	\triangle	$\heartsuit$	\hearsuit
∞	\infty	\	\backslash	$\spadesuit$	\spadesuit

1.1.2. Velké matematické operátory – třída 1

Velké matematické operátory jsou ve dvou velikostech: velké pro zobrazování matematiky a menší pro běžnou matematiku [1]:

$\sum$ $\sum$	\sum	$\cap$ $\bigcap$	\bigcap	$\odot$ $\bigodot$	\bigodot
$\prod$ $\prod$	\prod	$\cup$ $\bigcup$	\bigcup	$\otimes$ $\bigotimes$	\bigotimes
$\coprod$ $\coprod$	\coprod	$\sqcup$ $\bigsqcup$	\bigsqcup	$\oplus$ $\bigoplus$	\bigoplus
$\int$ $\int$	\int	$\vee$ $\bigvee$	\bigvee	$\uplus$ $\biguplus$	\biguplus
$\oint$ $\oint$	\oint	$\wedge$ $\bigwedge$	\bigwedge		

1.1.3. Binární operátory – třída 2

Binární operátory můžeme psát dvěma způsoby: $\cup$ nebo $\bigcup$ podle následujícího pravidla. Binární forma je používána mezi písmeny nebo výrazy $(A \cup B)$,

zatímco velká forma se používá, když je tam jen jeden výraz, obvykle s indexy, u operátorů ($\bigcup_i A$). Další problém může nastat u znaku `\setminus`, označující rozdíl. To dává stejný symbol jako `\backslash`, u kterého jsou patrné větší mezery obvyklé pro binární operace. K odstranění mezery můžeme použít binární operátor `$u{\circ}v$`, který nám dává na výstupu $u \circ v$ [1].

$+$	+	$-$	-	$\div$	\div
$\pm$	\pm	$\cap$	\cap	$\vee$	\vee nebo \lor
$\mp$	\mp	$\cup$	\cup	$\wedge$	\wedge nebo \land
$\setminus$	\setminus	$\uplus$	\uplus	$\oplus$	\oplus
$\cdot$	\cdot	$\sqcap$	\sqcap	$\ominus$	\ominus
$\times$	\times	$\sqcup$	\sqcup	$\otimes$	\otimes
$*$	\ast nebo *	$\triangleleft$	\triangleleft	$\oslash$	\oslash
$\star$	\star	$\triangleright$	\triangleright	$\odot$	\odot
$\diamond$	\diamond	$\wr$	\wr	$\dagger$	\dagger
$\circ$	\circ	$\bigcirc$	\bigcirc	$\ddagger$	\ddagger
$\bullet$	\bullet	$\triangle$	\bigtriangleup	II	\amalg
		$\triangledown$	\bigtriangledown		

1.1.4. Relace – třída 3

Relace nám vyjadřuje vlastnosti matematických objektů [1]:

$<$	<	$>$	>	$=$	=
$\leq$	\leq nebo \le	$\geq$	\geq nebo \ge	$\equiv$	\equiv
$\prec$	\prec	$\succ$	\succ	$\sim$	\sim
$\preceq$	\preceq	$\succeq$	\succeq	$\simeq$	\simeq
$\ll$	\ll	$\gg$	\gg	$\asymp$	\asymp
$\subset$	\subset	$\supset$	\supset	$\approx$	\approx
$\subseteq$	\subseteq	$\supseteq$	\supseteq	$\cong$	\cong
$\sqsubseteq$	\sqsubseteq	$\sqsupseteq$	\sqsupseteq	$\bowtie$	\bowtie
$\in$	\in	$\ni$	\ni	$\propto$	\propto
$\vdash$	\vdash	$\dashv$	\dashv	$\models$	\models
$\smile$	\smile	$\mid$	\mid	$\doteq$	\doteq
$\frown$	\frown	$\parallel$	\parallel	$\perp$	\perp

K negování relace se používá řídící slovo `\not` [1]:

$\not<$	\not<	$\not>$	\not>	$\neq$	\not=
$\not\leq$	\not\leq	$\not\geq$	\not\geq	$\not\equiv$	\not\equiv
$\not\prec$	\not\prec	$\not\succ$	\not\succ	$\not\sim$	\not\sim
$\not\preceq$	\not\preceq	$\not\succeq$	\not\succeq	$\not\simeq$	\not\simeq
$\not\subset$	\not\subset	$\not\supset$	\not\supset	$\not\approx$	\not\approx
$\not\subseteq$	\not\subseteq	$\not\supseteq$	\not\supseteq	$\not\cong$	\not\cong
$\not\sqsubseteq$	\not\sqsubseteq	$\not\sqsupseteq$	\not\sqsupseteq	$\notin$	\notin

Šipky jsou speciálním typem relací, které dělíme na horizontální a vertikální. Většina horizontálních šipek je dvojí velikosti, ty větší dostaneme pomocí předpony `\long`. Vertikální šipky také rostou, ale pomocí různých předpon. Příkaz `\iff` dává stejný symbol jako `\Longleftrightarrow` s tím rozdílem, že dává větší mezeru na obou stranách (je to vlastně symbolický ekvivalent výrazu „tehdy a jen tehdy“) [1].

$\leftarrow$	\leftarrow, \gets	$\longleftarrow$	\longleftarrow	$\uparrow$	\uparrow
$\rightarrow$	\rightarrow, \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		

1.1.5. Otevírací a zavírací závorky – třída 4 a 5

Závorky a další symboly, které píšeme v párech, jsou nazývány oddělovače, jak uvádí Michal Doob v knize Jemný úvod to TEXu [5]. Závorky sázíme prostřednictvím značek, nejsou totiž k dispozici na všech klávesnicích [1].

(	(	[	[nebo \lbrack	⌊	\lfloor
)	)	]	] nebo \rbrack	⌋	\rfloor
⟨	\langle	{	\{ nebo \lbrace	⌈	\lceil
⟩	\rangle	}	\} nebo \rbrace	⌉	\rceil

1.2. Písmo v matematickém režimu

Písmena v matematických vzorcích jsou ve shodě se sazbou nejkrásnějších matematických knih sázena pomocí kurzívy Computer Modern Math Italic. Toto písmo (font cmmi) nazýváme matematickou kurzívou.

Zde je nutné deklarovat tzv. matematickou rodinu a načíst font ve velikostech nutných pro matematickou sazbu. Je-li písmo velikosti n, pak první index má velikost $n-3$, druhý a další $n-5$. Např. pro sazbu oblíbeného matematického skriptu rsfs definujeme (viz např. [3], [6]):

```
\newfam\scrfam
\font\tenscr=rsfs10  \font\sevenscr=rsfs7  \font\fivescr=rsfs5
\textfont\scrfam=\tenscr
\scriptfont\scrfam=\sevenscr
\scriptscriptfont\scrfam=\fivescr
\def\scr #1{{\fam\scrfam #1}}.
```

Připojili jsme font rsfs v požadovaných velikostech, přiřadili ho „členům rodiny“ a definovali přepínač `\scr`. Ten v plainTEXu umožní matematickou sazbu tímto fontem: `$\scr{A}-{\fam0 B}=\scr{C}$` dá $\mathscr{A}-\mathrm{B}=\mathscr{C}$.

Základní rodiny matematických fontů

Petr Olšák ve své knize [3] rozebírá problém matematické sazby velmi podrobně. Bezpodmínečně musí být deklarovány rodiny číslo 2 a 3. Z fontů těchto rodin čerpá TEX veškeré znalosti o tom, jak sestavovat matematickou sazbu – jak vysoko posunout exponent nebo kolik místa udělat kolem zlomkové čáry.

- Rodina 0: Antikva; běžně v textu a méně v matematice (cos, lim apod.).

- Rodina 1: Kurzíva; sazba matematických proměnných (a, β, x, f apod).
- Rodina 2: Značky v matematice, binární operace a relace ($\odot$, $\leq$, atd.).
- Rodina 3: Operátory a „zvětšující se“ objekty pomocí následníků (např. $\sqrt{a}$, $\sqrt{\frac{a}{b}}$, $\sqrt{\frac{\frac{a}{c}}{b}}$).

Rodiny 0 až 3 jsou v plainu deklarovány takto [3]:

	\textfont	\scriptfont	\scriptscriptfont
Rodina 0:	cmr10 (\tenrm)	cmr7 (\sevenrm)	cmr5 (\fiverm)
Rodina 1:	cmmi10 (\teni)	cmmi7 (\seveni)	cmmi5 (\fivei)
Rodina 2:	cmsy10 (\tensy)	cmsy7 (\sevensy)	cmsy5 (\fivey)
Rodina 3:	cmex10 (\tenex)	cmex10 (\tenex)	cmex10 (\tenex)

V závorce je přepínač, který byl u každého fontu použit při zavedení primitivem `\font` (je možné ho použít při sazbě v textovém režimu), v matematickém režimu použijeme přepínač `\fam` následovaný číslem rodiny, např. `\fam0`. Pro rodinu 1 není použita běžná textová kurzíva cmti, ale speciální matematická. V rodině 3 máme ve všech třech variantách zaveden jediný font. Velké operátory se v matematických vzorcích používají v základní velikosti [3].

Každé rodině je přiřazena trojice fontů, které jsou zavedeny primitivem `\font`. První člen trojice (`\textfont`) určuje font základní velikosti, který TEX použije v matematických stylech D, D', T nebo T'. Druhý člen trojice (`\scriptfont`) je font s menšími znaky pro sazbu v indexové velikosti a TEX jej použije ve stylech S a S'. Třetí člen (`\scriptscriptfont`) obsahuje znaky pro sazbu indexů druhé úrovně a TEX ho použije ve stylech SS a SS' [3].

Ukázka fontů užívaných v matematické sazbě (použité makro poskytl vedoucí práce):

Font cmmi10:

0: Γ	1: Δ	2: Θ	3: Λ	4: Ξ	5: Π	6: Σ	7: Υ	8: Φ	9: Ψ
10: Ω	11: α	12: β	13: γ	14: δ	15: ϵ	16: ζ	17: η	18: θ	19: ι
20: κ	21: λ	22: μ	23: ν	24: ξ	25: π	26: ρ	27: σ	28: τ	29: υ
30: ϕ	31: χ	32: ψ	33: ω	34: ε	35: ϑ	36: ϖ	37: ϱ	38: ς	39: φ

40: $\leftharpoondown$ 41: $\leftharpoonup$ 42: $\rightharpoondown$ 43: $\rightharpoonup$ 44: $\lhook$ 45: $\rhook$ 46: $\triangleright$ 47: $\triangleleft$ 48: 0 49: 1

50: 2 51: 3 52: 4 53: 5 54: 6 55: 7 56: 8 57: 9 58: . 59: ,

60: $<$ 61: $/$ 62: $>$ 63: $\star$ 64: ∂ 65: A 66: B 67: C 68: D 69: E

70: F 71: G 72: H 73: I 74: J 75: K 76: L 77: M 78: N 79: O

80: P 81: Q 82: R 83: S 84: T 85: U 86: V 87: W 88: X 89: Y

90: Z 91: $\flat$ 92: $\natural$ 93: $\sharp$ 94: $\smile$ 95: $\frown$ 96: ℓ 97: a 98: b 99: c

100: d 101: e 102: f 103: g 104: h 105: i 106: j 107: k 108: l 109: m

110: n 111: o 112: p 113: q 114: r 115: s 116: t 117: u 118: v 119: w

120: x 121: y 122: z 123: $\imath$ 124: $\jmath$ 125: $\wp$ 126: $\vec{}$ 127: $\frown$

Kaligrafické písmo a některé symboly cmsy10:

0: $-$ 1: $\cdot$ 2: $\times$ 3: $*$ 4: $\div$ 5: $\diamond$ 6: $\pm$ 7: $\mp$ 8: $\oplus$ 9: $\ominus$

10: $\otimes$ 11: $\oslash$ 12: $\odot$ 13: $\bigcirc$ 14: $\circ$ 15: $\bullet$ 16: $\asymp$ 17: $\equiv$ 18: $\subseteq$ 19: $\supseteq$

20: $\leq$ 21: $\geq$ 22: $\preceq$ 23: $\succeq$ 24: $\sim$ 25: $\approx$ 26: $\subset$ 27: $\supset$ 28: $\ll$ 29: $\gg$

30: $\prec$ 31: $\succ$ 32: $\leftarrow$ 33: $\rightarrow$ 34: $\uparrow$ 35: $\downarrow$ 36: $\leftrightarrow$ 37: $\nearrow$ 38: $\searrow$ 39: $\simeq$

40: $\Leftarrow$ 41: $\Rightarrow$ 42: $\Uparrow$ 43: $\Downarrow$ 44: $\Leftrightarrow$ 45: $\nwarrow$ 46: $\swarrow$ 47: $\propto$ 48: $\prime$ 49: ∞

50: $\in$ 51: $\ni$ 52: $\triangle$ 53: $\triangledown$ 54: $/$ 55: $\imath$ 56: $\forall$ 57: $\exists$ 58: $\neg$ 59: $\emptyset$

60: $\Re$ 61: $\Im$ 62: $\top$ 63: $\bot$ 64: $\aleph$ 65: $\mathcal{A}$ 66: $\mathcal{B}$ 67: $\mathcal{C}$ 68: $\mathcal{D}$ 69: $\mathcal{E}$

70: $\mathcal{F}$ 71: $\mathcal{G}$ 72: $\mathcal{H}$ 73: $\mathcal{I}$ 74: $\mathcal{J}$ 75: $\mathcal{K}$ 76: $\mathcal{L}$ 77: $\mathcal{M}$ 78: $\mathcal{N}$ 79: $\mathcal{O}$

80: $\mathcal{P}$ 81: $\mathcal{Q}$ 82: $\mathcal{R}$ 83: $\mathcal{S}$ 84: $\mathcal{T}$ 85: $\mathcal{U}$ 86: $\mathcal{V}$ 87: $\mathcal{W}$ 88: $\mathcal{X}$ 89: $\mathcal{Y}$

90: $\mathcal{Z}$ 91: $\cup$ 92: $\cap$ 93: $\uplus$ 94: $\wedge$ 95: $\vee$ 96: $\vdash$ 97: $\dashv$ 98: $\lfloor$ 99: $\rfloor$

100: $\lceil$ 101: $\rceil$ 102: $\{$ 103: $\}$ 104: $\langle$ 105: $\rangle$ 106: $|$ 107: $\|$ 108: $\updownarrow$ 109: $\Updownarrow$

110: $\backslash$ 111: $\wr$ 112: $\surd$ 113: $\amalg$ 114: ∇ 115: $\int$ 116: $\sqcup$ 117: $\sqcap$ 118: $\sqsubseteq$ 119: $\sqsupseteq$

120: $\S$ 121: $\dagger$ 122: $\ddagger$ 123: $\P$ 124: $\clubsuit$ 125: $\diamondsuit$ 126: $\heartsuit$ 127: $\spadesuit$

Můžeme užívat i standardní písmo, pak text i s přepínačem vkládáme do horizontálního boxu `\hbox{}`, např. `$a+b=1\ \hbox{\sl přičemž $b<1$}$` dá $a+b=1$ *přičemž* $b<1$ [5].

Jméno písma	TEXový přepínač	Vzorek písma
Antikva/Roman	\rm	Antikva
Polutučně/Boldface	\bf	**Polotučně**
Kurzíva/Italic	\it	*Kurzíva*
Skloněné/Slanted	\sl	*Skloněné*
Stroj/Typewriter	\tt	`Strojopis`

Uvedené přepínače ovšem vyberou font v normální velikosti, a pokud chceme „nematematické“ písmo používat v horních a dolních indexech, musíme ho připojit v příslušné velikosti (10–7–5, nebo 12–9–7).

Ostatní písma: `\mit` se užívá pro kurzívu při psaní řeckých písmen (např. `${\mit\Pi}$` nám dává $\mit\Pi$). `\cal` se používá pro velká kaligrafická písmena (např. `${\cal ABCDEF}$` dává $\cal ABCDEF$); `\oldstyle` se používá pouze pro „skákavé“ číslice (např. `$\oldstyle 1234567890$` dává $\oldstyle 1234567890$) [5].

2. Sazba matematických formulí

2.1. Horní a dolní indexy

Pro dolní index se používá řídící znak _ a pro horní index ^. Pokud se dolní index skládá z více znaků, dáváme tuto skupinu do složených závorek {} [5]:

`$x_k+y_{k,j+1}$` ... $x_k + y_{k,j+1}$.

Indexy druhé úrovně nemůžeme psát `$y_k_j$`, protože tomu odpovídají dvě interpretace a TEX ohlásí chybu. Máme dvě možnosti: `$y_{k_j}$` a `${y_k}_j$`, dostaneme dva různé výsledky y_{k_j} a y_{k_j}. Obvykle používáme první možnost, ale záleží na typu publikace (učebnice) nebo přání designéra knihy. Stejné platí i pro horní indexy [1]:

`$B=2^{3^{4^{5}}}$` ... $B = 2^{3^{4^5}}$.

Horní a dolní indexy můžeme psát v libovolném pořadí, varianty `$x^2_0$` a `$x_0^2$` dávají stejný výsledek x_0^2 [5].

Usazení indexů u symbolů jsou řízena sekvencemi `\limits`, resp. `\nolimits`. První umisťuje dolní index pod symbol a horní nad symbol. Druhý umisťuje

dolní index u symbolu vpravo dole a horní index vpravo nahoře – tento způsob méně narušuje řádkovou osnovu, užívá se v horizontálním módu. Velké operátory se používají ve vysazeném módu v režimu `\limits`, resp. a v textovém módu v režimu `\nolimits` [3]:

`$\sum_{i=1}^\infty a_i$` ... $\sum_{i=1}^{\infty} a_i$

`$\sum_{i=1}\limits^\infty a_i$` ... $\sum\limits_{i=1}^{\infty} a_i$

`$\displaystyle\sum_{i=1}^\infty a_i$` ... $\displaystyle\sum_{i=1}^{\infty} a_i$

`$\displaystyle\sum\nolimits_{i=1}^\infty a_i$` ... $\displaystyle\sum\nolimits_{i=1}^{\infty} a_i$.

2.2. Sazba s akcentem

Matematické akcenty, neboli diakritická znaménka, vkládáme pomocí řídicích slov, která jsou uvedena v následující tabulce [1]:

$\hat{a}$	\hat a	$\check{a}$	\check a	$\tilde{a}$	\tilde a
$\grave{a}$	\grave a	$\acute{a}$	\acute a	$\vec{a}$	\vec a
$\dot{a}$	\dot a	$\ddot{a}$	\ddot a	$\breve{a}$	\breve a
		$\bar{a}$	\bar a		

Všechny tyto akcenty jsou určeny pro jednotlivé znaky. Na pokrytí více písmen se používá řídicí sekvence `\widehat{}` a `\widetilde{}` [1]:

`$\widehat{ab}$` ... $\widehat{ab}$

`$\widetilde{ab}$` ... $\widetilde{ab}$.

2.3. Matematické styly

Pomocí matematického stylu se nastavuje velikost písma a způsob indexování. Rozlišujeme tyto styly [3]:

Označení	Primitiv	Vysvětlení
D	`\displaystyle`	Základní velikost při psaní rovnic
T	`\textstyle`	Základní velikost v textu odstavce
S	`\scriptstyle`	Index a exponent první úrovně
SS	`\scripcsriptstyle`	Index a exponent druhé a další úrovně
D'		Redukované D
T'		Redukované T
S'		Redukované S
SS'		Redukované SS

Displaystyle : všechny znaky jsou v základní velikosti. Používají se v matematickém módu (mezi zdvojenými dolary) [1].

Textstyle : styl, ve kterém se sází matematika, je v textovém módu. Písmena mají stejnou velikost jako okolní text a píšeme je mezi jeden dolar [1].

Scriptstyle : pokud TEX vidí dolní nebo horní index, automaticky se přepne do scriptstylu [1].

Scriptscriptstyle : pro indexy druhé úrovně. Pokud je TEX ve scriptstylu, automaticky se přepne do scriptscriptstylu [1].

Scriptscriptstyle použije nejmenší velikost písma a žádné SSS už dále neexistuje. K přechodu mezi styly dochází při sazbě základního atomu a indexů [3]:

$$D^{S}_{S'}, \quad T^{S}_{S'}, \quad S^{SS}_{SS'}, \quad SS^{SS}_{SS'}, \quad {D'}^{S}_{S'}, \quad {T'}^{S}_{S'}, \quad {S'}^{SS}_{SS'}, \quad {SS'}^{SS}_{SS'}$$

Petr Olšák v knize TEXbook naruby říká [3]: „První vzorec znamená, že TEX ve stylu D při sazbě exponentu přechází do stylu S a při sazbě indexu do stylu S'.“

Při sazbě zlomků dochází také ke změně:

$$D\text{:}\ \frac{T}{T'}, \quad D'\text{:}\ \frac{T'}{T'}, \quad T\text{:}\ \frac{S}{S'}, \quad T'\text{:}\ \frac{S'}{S'}, \quad S, SS\text{:}\ \frac{SS}{SS'}, \quad S', SS'\text{:}\ \frac{SS'}{SS'}$$

„První údaj nám říká, že ve stylu D je čitatel sázen stylem T a jmenovatel stylem T'.“ [3] .

Velikost velkých operátorů není jediný rozdíl mezi displaystyle a textstyle. V textstyle TEX omezuje výšku a hloubku vzorců, takže nebude narušena řádková osnova. Z tohoto důvodu, jsou-li na sebe svisle napojeny např. zlomky, zvolí se scriptstyle. Rozdíl vidíme v následující ukázce [1]:

`$n+{m+1 \over 1+n^2}$` ... $n+\frac{m+1}{1+n^2}$

`$\displaystyle n+{m+1 \over 1+n^2}$` ... $n+\displaystyle\frac{m+1}{1+n^2}$.

V prvním případě začíná TEX v textstyle a přechází do scriptstyle pro čitatele a jmenovatele zlomku, pro exponent 2 je použit scriptscriptstyle. Měnit styly můžeme příkazem `\displaystyle`, `\textstyle`, `\scriptstyle` nebo `\scriptscriptstyle`. Změna zůstane v platnosti až do konce aktuální skupiny [1].

2.4. Sazba matematických objektů

2.4.1. Mezery v matematické sazbě

Bylo uvedeno, že TEX v matematické sazbě používá automatické mezerování dané třídou znaku. Pokud nám toto mezerování nevyhovuje, nastavíme si ho pomocí řídících sekvencí sami. Použijeme `\kern`, `\hskip`, `\hfil` nebo `\mkern`, `\mskip`. Při použití `\kern` a `\hskip` můžeme pro nastavení velikosti mezery použít libovolnou dimenzi. Na rozdíl od toho v případu `\mkern`, `\mskip` se nastavuje velikost mezery pomocí jednotky mu, pro kterou platí 18 mu = velikost použitého písma. Jednotka mu tak závisí na matematickém stylu, ve kterém byla použita [3].

2.4.2. Sazba názvů matematických funkcí

Matematické funkce jsou sázeny vzpřímeným písmem – roman. Není použita kurzíva jako při sazbě veličin. V následující tabulce jsou uvedené řídící sekvence pro předdefinované funkce [2]:

\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

2.4.3. Zlomky

Zlomky se sází příkazem `\over`, který odděluje čitatele od jmenovatele. Pro přesné vymezení je vkládáme do skupiny (do složených závorek). Použití ukážeme na příkladu [1]:

```
$${x+y+z\over x-z} \quad\hbox{a}\quad { x+y+{z\over x}-z}$$
```

dává

$$\frac{x+y+z}{x-z} \quad \text{a} \quad x+y+\frac{z}{x}-z.$$

V případě, že TEX vidí zlomek v displaystyle, nastaví jeho složky v textstyle. Je-li v textovém stylu, vyjdou komponenty ve scriptstyle, a pokud je ve scriptstyle, komponenty se nastaví do scripscriptstyle [1].

Pokud dáváme přednost $\frac{x}{x+y}$ před $\frac{x}{x+y}$, musíme to explicitně uvést pomocí příslušného přepínače [1].

Zlomek lze psát dvěma styly: lze to vysázet buď `$x/z$`, což nám na výstupu dá x/z nebo pomocí řídícího znaku `\over`, jehož použití jsme viděli dříve. První způsob často nabízí lepší alternativu pro psaní zlomku v textu [1].

2.4.4. Řetězové zlomky

Tento zlomek vyžaduje použití přepínače `\displaystyle`.

Standardní způsob psaní zlomku:

```
$$ a=a_0+{1\over a_1+{1\over a_2+{1\over a_3}}}$$
```

dává

$$a = a_0 + \frac{1}{a_1 + \frac{1}{a_2 + \frac{1}{a_3}}}$$

Vidíme, že velikost znaků postupně klesá, což v případě řetězového zlomku není správné.

Jiný způsob:

```
$$a=a_0+{1\over\displaystyle a_1+{1\over\displaystyle a_2+
{1\over\displaystyle a_3}}}$$
```

vysází

$$a = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3}}}$$

Opět to není správně, všimněme si, že 1 je příliš blízko zlomkové čáře.

Třetí způsob: PlainTEX definuje pro matematické formule velikost a hloubku výrazů v závorce pomocí příkazu `\mathstrut`:

```
$$a=a_0+{1\over\displaystyle a_1+
{\mathstrut 1\over\displaystyle a_2+
{\mathstrut 1\overdisplaystyle a_3}}}$$
```

Tak dostaneme

$$a = a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3}}}$$

Odstraňuje to nedostatek druhého způsobu [1].

2.4.5. Odmocniny a mocniny

Odmocniny lze vysázet dvěma příkazy: `\sqrt` nebo `\root`. Odmocnina se přizpůsobuje velikosti odmocněnce.

Např.

```
$$x=x^{3}\times{(\sqrt{x}+\sqrt{y})\over\sqrt{xy}}
\times{x^{2}\over\sqrt{x+\sqrt{y}}}
\times{y^{2}\over\sqrt{x\sqrt{y+\sqrt{x}}}}$$
```

dá

$$x = x^3 \times \frac{(\sqrt{x}+\sqrt{y})}{\sqrt{xy}} \times \frac{x^2}{\sqrt{x+\sqrt{y}}} \times \frac{y^2}{\sqrt{x\sqrt{y+\sqrt{x}}}}$$

Pokud pod odmocninou máme různě velké znaky, nemusí odmocniny vypadat stejně:

`$\sqrt c +\sqrt d+\sqrt{1+y_i^4}$` ... $\sqrt{c}+\sqrt{d}+\sqrt{1+y_i^4}$.

Pokud se nám výsledek nelíbí, použijeme příkaz `\mathsrut`, který dodá svoji výšku a hloubku:

`$\sqrt{\mathstrut c}+...$`... $\sqrt{c}+\sqrt{d}+\sqrt{1+y_i^4}$.

Příkaz `\root` umožňuje sázet n-tou odmocninu. Například:

`$\root\scriptstyle n\of{x+2}$` ... $\root\scriptstyle n\of{x+2}$

`$\displaystyle\root\scriptstyle n\of{x+2}$` ... $\displaystyle\root\scriptstyle n\of{x+2}$ [1].

Samozřejmě můžeme výšku a hloubku odmocniny nastavit libovolně užitím a to neviditelné `\vrule`. Např.

```
$$\sqrt{\vrule height25pt depth25pt width0pt
\vrule height0pt depth0pt width50pt}$$
```

dá

$$\sqrt{\vrule height25pt depth25pt width0pt \vrule height0pt depth0pt width50pt}$$

2.4.6. Horizontální oddělovače

Chceme-li umístit vodorovnou linku nad nebo pod matematický výraz, použijeme `\overline{...}` nebo `\underline{...}`:

`$\overline{a+c'}=\bar a+\bar{c'}$` ... $\overline{a+c'}=\bar{a}+\bar{c'}$.

Výrazy můžeme také podtrhávat:

`$\underline{a+b}$` ... $\underline{a+b}$.

Šipky získáme pomocí příkazu \overrightarrow a \overleftarrow. Opět je velký rozdíl mezi nimi a \vec, který je určen pouze pro jedno písmeno:

`$\overrightarrow{UV}+\vec{XY}$` ... $\overrightarrow{UV}+\vec{XY}$.

Na rozdíl od `\bar` a `\vec`, ostatní příkazy použité v této podkapitole neberou v úvahu sklon písma, který využívá kurzíva. Pro lepší výsledky si můžete představit ruční korekce s definicí, jako je:

```
\def\ora#1{\overrightarrow{\mkern-2mu#1\mkern 2mu}}
$\overrightarrow{XY}$, $\ora{XY}$
```
... $\overrightarrow{XY}$, $\ora{XY}$.

Příkazy `\overbrace` a `\underbrace` nám umisťují závorky nad a pod formuli:

`$\overbrace{a+\cdots+a}^{k\times}$` ... $\overbrace{a+\cdots+a}^{k\times}$

`$\underbrace{a+a+\cdots+a}_{n\times a}$` ... $\underbrace{a+a+\cdots+a}_{n\times a}$ [1].

2.4.7. Vertikální oddělovače

Existují dva způsoby, jak měnit velikost oddělovače. Buď vybíráme velikost sami (`\big` atd.), anebo necháme vhodnou velikost vybrat TEX podle obsahu mezi oddělovači `\left` a `\right`.

Vybíráme-li velikost oddělovače sami, máme k dispozici příkazy `\big`, `\Big`, `\bigg` a `\Bigg`. Jejich analogie s přidaným „l“ a „r“ lze použít jako levé, resp. pravé oddělovače, existuje i varianta s „m“ oddělující znaky vlevo a vpravo. Např.

$$\quad\Biggl(\quad\biggl(\quad\Bigl(\quad\bigl(\quad(\quad) \quad\bigr)\quad\Bigr)\quad\biggr)\quad\Biggr)$$

je vytvořeno pomocí

```
$$ \quad\Biggl(\quad\biggl(\quad\Bigl(\quad\bigl( \quad(\quad
   ) \quad\bigr)\quad\Bigr)\quad\biggr)\quad\Biggr) $$.
```

Velikost první značky `\big` je 12pt, používáme-li k sazbě 12bodové písmo, pak mezi velikostí `\big` a prostého znaku není rozdíl. Toto musíme mít stále na zřeteli.

`$h(x)=g\bigl(f(x)\bigr)$` ... $h(x)=g\bigl(f(x)\bigr)$

`$\bigl||y|+|1|\bigr|$` ... $\bigl||y|+|1|\bigr|$.

Přidané mezery ale výslednou sazbu mění. Srovnejme:

`$\bigl\{a+b\bigm| a\in A, b\in B\bigr\}$` ... $\{a+b \mid a \in A, b \in B\}$
s verzí bez `\bigm`

`$\bigl\{x+y| x\in X, y\in Y\bigr\}$` ... $\{x+y|x \in X, y \in Y\}$
a vidíme, že výsledek je rozdílný [1].

Značky pro výběr velikosti oddělovače lze samozřejmě použít u všech vertikálních oddělovačů jako jsou [2]:

$$[]\{\}\lfloor\rfloor\lceil\rceil\langle\rangle/\backslash\|\Uparrow\uparrow\Downarrow\downarrow\Updownarrow\updownarrow$$

Dále budeme zvětšovat velikost pomocí příkazu `\big`:

$$\big[\big]\big\{\big\}\big\lfloor\big\rfloor\big\lceil\big\rceil\big\langle\big\rangle\big/\big\backslash\big\|\big\Uparrow\big\uparrow\big\Downarrow\big\downarrow\big\Updownarrow\big\updownarrow$$

pomocí příkazu `\Bigl` a `\Bigr`:

$$\Big[\Big]\Big\{\Big\}\Big\lfloor\Big\rfloor\Big\lceil\Big\rceil\Big\langle\Big\rangle\Big/\Big\backslash\Big\|\Big\Uparrow\Big\uparrow\Big\Downarrow\Big\downarrow\Big\Updownarrow\Big\updownarrow$$

pomocí příkazu `\biggl` a `\biggr`:

$$\bigg[\bigg]\bigg\{\bigg\}\bigg\lfloor\bigg\rfloor\bigg\lceil\bigg\rceil\bigg\langle\bigg\rangle\bigg/\bigg\backslash\bigg\|\bigg\Uparrow\bigg\uparrow\bigg\Downarrow\bigg\downarrow\bigg\Updownarrow\bigg\updownarrow$$

a nakonec v největší velikosti pomocí příkazu `\Biggl` a `\Biggr`:

$$\Bigg[\Bigg]\Bigg\{\Bigg\}\Bigg\lfloor\Bigg\rfloor\Bigg\lceil\Bigg\rceil\Bigg\langle\Bigg\rangle\Bigg/\Bigg\backslash\Bigg\|\Bigg\Uparrow\Bigg\uparrow\Bigg\Downarrow\Bigg\downarrow\Bigg\Updownarrow\Bigg\updownarrow$$

Oddělovač `\bigr` můžeme používat samostatně bez `\bigl` a naopak. To však nelze u příkazů `\left` a `\right`, které fungují pouze v páru `\left(...\right)` [1]:

`$\displaystyle\left({\pi\over 4}+2k\pi\right)$` ... $\left(\frac{\pi}{4}+2k\pi\right)$

`$\left|{a\over x^2-y^2}\right|$` ... $\left|\frac{a}{x^2-y^2}\right|$.

Příkazy `\left` a `\right` jsou schopny vytvářet oddělovače o libovolné velikosti, například závorky kolem velké matice. Oddělovače `\left` a `\right` pracují pouze v páru a musí být oba ve stejné skupině, nemůžeme je používat takto:

`left(...&...\right)` nebo `\left(...{...\right)...}`. Pokud máme v jednom vzorci více oddělovačů `\left` a `\right`, musí být správně vnořeny. Oddělovače nemusí být stejné. Bez problémů se spáruje `\left(` s `\right]`. Můžeme dokonce spárovat `\left` s libovolným oddělovačem a `\right`, který vysází „neviditelný oddělovač“ [1]. Například vzorec [4]:

$$f(x) = \begin{cases} \frac{x}{x+1}, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

je vysázen sekvencí

```
$$ f(x)= \left\{ \matrix{
{x \over x+1}, & x \geq 0 \cr
0, & x<0 \cr}
\right $$.
```

Analogické platí pro `\left.`

Bohužel, nelze se spoléhat pouze `\left` a `\right` a vynechat ze sazečského repertoáru příkazy typu `\big`. TEX nevybírá pokaždé správnou velikost oddělovače, a proto ji musí sazeč v některých případech nastavit sám.

Jako další příklad uvedeme $\Big||y| + |1|\Big|$. Pokud místo `\Bigl...\Bigr` použijeme `\left...\right`, dostaneme $||y| + |1||$, což jsme asi nepotřebovali, ale TEX správně nastavil výšku oddělovače | podle výšky vnitřního boxu a ta je dána, jak vidíme, právě výškou vnitřní |, tj. „automatický“ oddělovač má výšku stejnou.

Problémy mohou také vzniknout, pokud výraz obsahuje velký operátor a limitu, např.

`$\displaystyle\left(\sum_{k=0}^n {1 \over k}\right) $` ... $\displaystyle\left(\sum_{k=0}^n {1 \over k}\right)$

`$\displaystyle\biggl(\sum_{k=0}^n {1 \over k}\biggr)$` ... $\displaystyle\biggl(\sum_{k=0}^n {1 \over k}\biggr)$

Symboly popsané v této části jsou vertikálně zarovnané na střed účaří[1]. To znamená, že rostou nahoru i dolů o stejnou velikost. Pokud máme vzorec, který je

[1] Je to myšlená linka, na níž se umísťují jednotlivé znaky, angl. baseline.

vysoký, ale ne dostatečně hluboký, budeme mít oddělovač příliš velký a spoustu prázdného místa. Naštěstí se to běžně nestává, protože velké operátory nebo zlomky mají tendenci se rozprostřít přibližně symetricky nad i pod účaří. Pokud se to stane, musí si sazeč umět poradit [1].

2.4.8. Symboly nad a pod objektem

Pokud chceme napsat text nad relací, můžeme použít příkaz `\bbuildrel`, který ovšem musíme sami definovat:

```
\def\bbuildrel#1_#2^#3{\mathrel{\mathop{\kern0pt#1}%
     \limits_{#2}^{#3}}}.
```

Příklad:

```
$$P_x(B) \bbuildrel=_{}^{\hbox{\sevenrm def}} P(X \in B)$$
```

dává:

$$P_x(B) \stackrel{\text{def}}{=} P(X \in B)$$

Při použití je nutné dodržet syntax i počet parametrů, nechceme-li některý index sázet, uvedeme parametr prázdný jako v našem příkladu. Znaky _ ^ slouží k vymezení argumentu. Pořadí, ve kterém jsou uvedeny, musíme zachovat [1].

2.4.9. Další relace

Tato podkapitola vychází z knihy A Beginner's Book of TEX[1]. Předpokládejme, že potřebujeme symbol pro novou relaci a musíme se rozhodnout, čím bychom mohli nakombinovat plus, kruh a šipku: $+\circ\rightarrow$. Tuto kontrolní sekvenci si nadefinujeme a nazveme ji např.: `\toto`

```
\def\relplus{\mathrel+} \def\relcirc{\mathrel\circ}
\def\totosymb{\relplus\joinrel\relcirc\joinrel\rightarrow}
\def\toto{\mathrel{\totosymb}}
```

Začneme tím, že změníme symboly v relaci (`\rightarrow` je již jeden): tímto stylem TEX mezi ně nedává prostor a jsou psány jeden po druhém. Zde je ukázka s mezerami: $+\circ \rightarrow$. Pak můžete použít `\joinrel` a dát znaky dohromady. Je to dobré opatření, protože znaky nemusí sahat tak daleko doprava a doleva jako jejich box hranice.

Pro display math můžeme vybudovat delší verzi: $+\!\!\circ\!\!\longrightarrow$

```
\def\relminus{\mathrel-}
\def\Totosymb{\relplus\joinrel\relminus
\joinrel\relcirc\joinrel\longrightarrow}
\def\Toto{\mathrel{\Totosymb}}
```

Chceme-li umístit něco nad a pod nový symbol, můžeme použít makra z předchozí části:

`$X \toto_{n\to\infty} Y$` ... $X \toto_{n\to\infty} Y$

`$$ X \bbuildrel \Toto_{n\to\infty}^{} Y$$`

$$X \bbuildrel \Toto_{n\to\infty}^{} Y$$

2.4.10. Limity

Limita je matematická operace, pro kterou se používá zkratka `\lim`:

`$\lim_{x\rightarrow 0} {\sin x \over x}=1$` ... $\lim_{x\to 0} \frac{\sin x}{x} = 1$

`$$\lim_{x\rightarrow 0} {\sin x \over x}=1$$`

$$\lim_{x\to 0} \frac{\sin x}{x} = 1$$

PlainTEX rovněž nabízí příkazy `\limsup` a `\liminf`, které se vytisknou jako lim sup a lim inf.

$$\lim_{n\to\infty}(a_n + b_n) \text{existuje} \iff \limsup_{n\to\infty}(a_n + b_n) = \liminf_{n\to\infty}(a_n + b_n).$$

Někdo dává raději přednost těmto alternativním zkratkám $\overline{\lim}$ a $\underline{\lim}$:

`\def\limsup{\mathop{\overline{\rm lim}}}`

`\def\liminf{\mathop{\underline{\rm lim}}}`.

K dispozici jsou také induktivní a projektivní limity zastoupené zkratkami lim ind a lim proj nebo $\varinjlim$ a $\varprojlim$. V plainTEXu není ani jeden předdefinován. První dvojici lze definovat takto:

```
\def\limind{\mathop{\rm lim\,ind}}
\def\limproj{\mathop{\rm lim\,proj}}.
```

Definovat druhou dvojici je poněkud složitější, protože nemáme žádné značky `\underrightarrow` a `\underleftarrow` pro umístění rozšiřitelných šipek pod výraz. Mohli bychom však využít výplní `\rightarrowfill` či `\leftarrowfill`:

```
\def\limr{\mathop{\mathop{\rm lim}\limits_{\hbox{\rightarrowfill}}}}
\def\liml{\mathop{\mathop{\rm lim}\limits_{\hbox{\leftarrowfill}}}}.
```

Potom

`$\displaystyle \liml_{n\to\infty} a$ \qquad $\limr_{n\to\infty} a$`

dá tento výsledek: $\displaystyle \liml_{n\to\infty} a \qquad \limr_{n\to\infty} a$

Dobrou ideou je také využití značek `\longrightarrow` a `\longleftarrow`, které poskytují víceméně správnou délku. Makro `\oalign` (což je v podstatě `\halign` s triviální jednosloupcovou šablonou `#\cr` a prokladem nastaveným na minimum) přiblíží šipku co nejvíce k limitě.

```
\def\limind{\mathop{\oalign{\hfil$\rm lim$\hfil\cr
    $\longrightarrow$\cr}}}
\def\limproj{\mathop{\oalign{\hfil$\rm lim$\hfil\cr
    $\longleftarrow$\cr}}}
```

`$S=\limind_{\,\gamma\in G}S_\gamma$` ... $S=\limind_{\,\gamma\in G}S_\gamma$

`$$S=\limind_{\,\gamma\in G}S_\gamma$$` ... $$S=\limind_{\,\gamma\in G}S_\gamma$$

`$W=\limproj_{\,\eta\in Z}W_\eta$` ... $W=\limproj_{\,\eta\in Z}W_\eta$

`$$W=\limproj_{\,\eta\in Z}W_\eta$$` ... $$W=\limproj_{\,\eta\in Z}W_\eta \quad [1]$$

2.5. Fantóm – prázdný box

Příkazy `\phantom`, `\vphantom` a `\hphantom` vytváří prázdné boxy (neviditelné), které mají rozměry dané jejich parametrem. Připomeňme, že `\strut` má nulovou šířku, ale nenulovou výšku a hloubku. Fantómy se používají např. pro zarovnávání textu, zajištění konzistentní vzdálenosti pod odmocninou a u zlomkové čáry aj. V plainTEXu je předdefinován `\strut` vysoký 8,5 pt a 3,5 pt hluboký a `\mathsrut`, který je přesně tak vysoký a hluboký jako kulaté závorky. Pro srovnání uvedeme příklad s `\vrule`:

`( \hbox{\mathstrut\vrule} ) \hbox{\strut\vrule}` ... (|) |

Makro `\vphantom{formule}` přečte formuli v parametru a vysází prázdný box s výškou a hloubkou danou materiálem v parametru (s nulovou šířkou). Příkaz `\vphantom` můžeme použit v jakémkoli režimu.

Podobně, `\hphantom` má nulovou výšku a hloubku, ale šířku má rovnu šířce materiálu v parametru. Opět ho nemusíme používat jen v matematickém módu [1].

2.5.1. Výpustky

Výpustku znázorňujeme třemi po sobě následujícími tečkami. Napíšeme-li tři tečky, TEX samozřejmě vysází tři tečky, ale budou příliš blízko u sebe ... Pro výpustku požíváme příkaz `\dots` – ... V matematice existují i jiné značky pro výpustku:

- `\ldots` dává nízké tečky, jako je `\dots`. Používá se mezi čárkami a interpunkčními znaménky:

  `$a=(a_1,\ldots,a_n)$` ... $a = (a_1, \ldots, a_n)$

- `\cdots` dává tečky na úrovni znaménka +, vypadá nejlépe mezi operátory, jako je + a ×:

  `$A=a_1+\cdots+a_n$` ... $A = a_1 + \cdots + a_n$

- `\vdots`, resp. `\ddots` tečky uspořádá svisle, resp. diagonálně. Používáme je např. v maticích:

```
$$A=\pmatrix{
a_{11}&a_{12}&\cdots&a_{1n}\cr
a_{21}&a_{22}&\cdots&a_{2n}\cr
\vdots&\vdots&\ddots&\vdots\cr
a_{m1}&a_{m2}&\cdots&a_{mn}\cr
}$$
```

$$A=\pmatrix{a_{11}&a_{12}&\cdots&a_{1n}\cr a_{21}&a_{22}&\cdots&a_{2n}\cr \vdots&\vdots&\ddots&\vdots\cr a_{m1}&a_{m2}&\cdots&a_{mn}\cr}$$

Někdy potřebujeme diagonální tečky v opačném směru, než který plainTEX nabízí. Můžeme využít analogie s definicí `\ddots` ze strany 359 TEXbooku a novou značku nazvat `\adots`:

```
\def\adots{\mathinner{\mkern2mu\raise1pt\hbox{.}\mkern2mu
     \raise4pt\hbox{.}\mkern2mu\raise7pt\hbox{.}\mkern1mu}}

   $\adots\ddots\ldots\adots\ddots$
```

$\adots\ddots\ldots\adots\ddots$ [1].

3. Sazba víceřádkových formulí

Pokud chceme vysázet vycentrovaný vzorec na samostatném řádku, provedeme to ve vysazeném módu – dáme ho mezi dvojité dolary $$. Sázet takto několik vzorců pod sebou se sice může někdy vyplatit, ve většině případů, ale výsledek nebude pěkný. Mezi jednotlivými řádky budou velké mezery. Proto je lepší použít makra pro sazbu víceřádkových formulí.

PlainTEX nabízí za tímto účelem příkaz `\displaylines`. Konstrukce

```
$$\openup2pt \displaylines{
\cal B(\alpha,\beta)=\int^1_0{x^{\alpha-1}(1-x)^{\beta-1}}dx \cr
\cal B(\alpha,\beta)=\cal B(\beta,\alpha) \cr
\cal B(\alpha,\beta)={\Gamma(\alpha)\Gamma(\beta)
    \over \Gamma(\alpha+\beta)}  \cr }$$
```

vede k výsledku

$$\mathcal{B}(\alpha,\beta)=\int_0^1 x^{\alpha-1}(1-x)^{\beta-1}dx$$

$$\mathcal{B}(\alpha,\beta)=\mathcal{B}(\beta,\alpha)$$

$$\mathcal{B}(\alpha,\beta)=\frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$$

Na co si musíme dát pozor:

- Vzorec, který se nevejde na jeden řádek, TEX sám od sebe nerozdělí.
- Každý řádek musí být ukončen `\cr`.
- Jakákoliv interpunkce musí být napsána před `\cr`. Pokud ji uvedeme až po `\cr`, objeví se na začátku dalšího řádku.
- Častou chybou je zapomenutí závorek, které uzavírají `\displaylines`. Proto je nejlepší začít prázdnou strukturou

  ```
  \displaylines{
   } $$
  ```

  a potom vyplnit vnitřek.
- Obsahují-li řádky velké operátory nebo zlomky, je potřeba pro lepší vizuální efekt zvětšit proklad. Proto je před `\displaylines` uvedeno `\openup2pt`.

Struktura `\displaylines` centruje řádky pomocí slabé pružné mezery `\hfil`. Toho lze využít k umístění textu na řádku podle vlastní představy – použitím více značek `\hfil` nebo silnějšího `\hfill` [1]:

```
$$\displaylines{
\qquad V=1a_1+2a_2+\cdots+Ta_T= \hfill\cr
(a_1+a_2+\cdots+a_T)+(a_2+\cdots+a_T)+\cdots+a_T= \cr \hfill
r_0+r_1+\cdots+r_{T-1} \qquad\cr
}$$
```

dá

$$V = 1a_1 + 2a_2 + \cdots + Ta_T = \\ (a_1 + a_2 + \cdots + a_T) + (a_2 + \cdots + a_T) + \cdots + a_T = \\ r_0 + r_1 + \cdots + r_{T-1}$$

3.1. Zarovnání víceřádkových formulí

Makro `\eqalign` slouží pro sázení několika formulí pod sebou do řádků. Symbol & určuje, kde budou rovnice zarovnány. Je zvykem zarovnávat podle znaku rovnosti, ale není to pravidlem. Každý řádek končíme příkazem `\cr`. Například:

```
$$ \eqalign{
x + y &= z \cr
12x + 2y - 5z &= 12 \cr
xy &= -5 \cr
} $$
```

$$\eqalign{
x + y &= z \cr
12x + 2y - 5z &= 12 \cr
xy &= -5 \cr
}$$

Levá strana jednoho nebo více vzorců s `\eqalign` může být prázdná:

```
$$\eqalign{
(x+1)^4-(x-1)^2 &=x^4+4x^3+6x^2+4x+1-(x^2-2x+1)\cr
&=x^4+4x^3+5x^2+6x \cr}
$$
```

$$\eqalign{
(x+1)^4-(x-1)^2 &=x^4+4x^3+6x^2+4x+1-(x^2-2x+1)\cr
&=x^4+4x^3+5x^2+6x \cr}$$

Pokud na řádcích není nic před &, formule budou zarovnány doleva. Pravé centrování jde stejně snadno. Makro `\eqalign` lze využít i mimo matematiku.

Příklad: Nejprve napíšeme text (citaci) a jméno autora do paměťových boxů:

```
\setbox1=\vbox{\hsize3.3in \noindent \uv{Jedině děti vědí,
          co hledají}, pravil...}
\setbox2=\vbox{\hbox{Antoine de}\hbox{Saint-Exup\'ery}}.
```

Pak vše zapíšeme v matematickém režimu [1]:

```
$$ \openup 6pt
\eqalign{&\left.\vcenter{\box1}\ \right\}\vcenter{\box2}\cr}
$$
```

$$\left.\begin{aligned}&\text{„Jedině děti vědí, co hledají“, pravil malý princ. „Ztrácejí čas pro hadrovou panenku, panenka}\\&\text{se stane hrozně důležitá, a když jim ji někdo vezme, pláčou...“ „Mají štěstí“, řekl výhybkář.}\end{aligned}\right\}\ \text{Antoine de Saint-Exupéry}$$

3.2. Značení vzorců – číslování rovnic

Rovnice číslujeme pomocí příkazů `\eqno` na pravý okraj a pomocí `\legno` na levý okraj. Například: `$$ a+b=c \eqno{(1)} $$`

$$a+b=c \eqno{(1)}$$

nebo `$$ a^2+b=c^3 \leqno{(1)} $$`

$$a^2+b=c^3 \leqno{(1)}$$

Jak vidíme, oba příkazy se používají analogicky.

Jedinou víceřádkovou formuli značíme vpravo za posledním řádkem nebo vlevo před prvním řádkem.

Chceme-li označit více formulí, můžeme kombinovat `\eqalign` s `\eqno`. Obvykle označené vzorce ohraničíme velkou svorkou.

```
$$
\openup2pt \left. \eqalign{
{\cal B}(\alpha,\beta)&=\int^1_0{x^{\alpha-1}(1-x)^{\beta-1}}dx\cr
{\cal B}(\alpha,\beta)&={\cal B}(\beta,\alpha) \cr
{\cal B}(\alpha,\beta)&={\Gamma(\alpha)\Gamma(\beta)
\over \Gamma(\alpha+\beta)}\cr}   \right\} \eqno{(3)}
$$
```

$$\left.\begin{aligned}\mathcal{B}(\alpha,\beta)&=\int_0^1 x^{\alpha-1}(1-x)^{\beta-1}dx\\ \mathcal{B}(\alpha,\beta)&=\mathcal{B}(\beta,\alpha)\\ \mathcal{B}(\alpha,\beta)&=\frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}\end{aligned}\right\} \tag{3}$$

Potřebujeme-li označit více formulí samostatně, použijeme `\eqalignno` nebo `\leqalignno`:

$$a_1+a_2+a_3=0 \tag{1}$$

$$3a_3=21 \tag{2}$$

$$12a_1+24a_2=36 \tag{3}$$

```
$$\eqalignno{
a_1+a_2+a_3&=0& (1) \cr
3a_3&=21& (2) \cr
12a_1+24a_2&=36&(3)\cr
}$$
```

Na každém řádku jsou nyní dva znaky: &&. Jeden určuje, kde se rovnice zarovná, a druhý slouží pro vymezení popisku. Nahrazením na začátku `\eqalignno` za `\legaligno`, dostaneme zrcadlové uspořádání [1]:

$$\text{(1)}\qquad a_1+a_2+a_3=0$$

$$\text{(2)}\qquad 3a_3=21$$

$$\text{(3)}\qquad 12a_1+24a_2=36$$

3.3. Matice

Matice se sází příkazem `\matrix`, kde se jednotlivé položky na řádku oddělují pomocí & a jednotlivé řádky jsou odděleny řídicím slovem `\cr`. Pokud řádky mají různé počty &, matice bude mít tolik sloupců, kolik jich má nejdelší řádek a ty kratší budou mít prázdné položky, jako by byly ukončeny `...&&&\cr`. Matice se zadávají ve tvaru `$$\matrix{...}$$`, kde se mezi složené závorky vypisují jednotlivé řádky matice.

```
$$ \matrix{
a & b & c \cr
\alpha & \beta & \gamma \cr
1 & 0 & 1 \cr } $$
```

$$\matrix{
a & b & c \cr
\alpha & \beta & \gamma \cr
1 & 0 & 1 \cr }$$

Každá položka matice tvoří skupinu a nemá vliv na jiné položky. Můžeme také přidat text mezi řádky pomocí `\noalign`. (Ale `\openup` a `\offinterlineskip` nebude fungovat, protože `\matrix` vrátí mezery na výchozí hodnotu.) Vstupní šablony jsou v podstatě typu `\hfil$#$\hfil`, aby každá položka byla přečtena v matematickém režimu a ve svém sloupci se vycentrovala. Budeme se dopouštět méně chyb, pokud při zadávání matice začneme se závorkami a teprve pak vyplníme vnitřek. Je také dobré sloupce zarovnávat rovnou ve zdrojovém souboru – záznamy jsou čteny v matematickém režimu a nehrozí nebezpečí rušivých mezer na výstupu.

Matice v závorkách

Matice jsou obvykle zadávány v kulatých závorkách. Toho docílíme pomocí `\pmatrix`, nemusíme psát `\left(\matrix{...}\right)`.

```
$$M=\pmatrix{
\pmatrix{\lambda & 1\cr
1 &\lambda\cr} &\bf \lambda\cr
\bf 1&\pmatrix{1 & 0 \cr
\lambda & 1\cr}
\cr}$$
```

$$M=\begin{pmatrix}\begin{pmatrix}\lambda & 1\\ 1 & \lambda\end{pmatrix} & \lambda\\ \mathbf{1} & \begin{pmatrix}1 & 0\\ \lambda & 1\end{pmatrix}\end{pmatrix}$$

TEX automaticky centruje malé matice vnořené v matici M.

Determinanty

Determinant má na rozdíl od matice místo závorek vertikální „pruhy“, které se snadno zkonstruují `\left|...\right|`:

$$\det(A)=\begin{vmatrix}a_{11} & \lambda & a_{13}\\ \lambda & a_{22} & a_{23}\\ a_{31} & a_{32} & \lambda\end{vmatrix}$$

Soustavy rovnic

I pro sazbu soustavy rovnic můžeme využívat `\matrix`:

```
$$\left\{\matrix{
\hfill 2x &+&\hfill 4y &+& 6z &=& 4\cr
-24x &-& 3y &-& \hfill z &=& 0 \cr
& & \hfill y^2 &+& \hfill 3z &=& 30 \cr}\right.
\leqno{(\Sigma)} $$
```

$$\left\{\matrix{
\hfill 2x &+&\hfill 4y &+& 6z &=& 4\cr
-24x &-& 3y &-& \hfill z &=& 0 \cr
& & \hfill y^2 &+& \hfill 3z &=& 30 \cr}\right. \qquad (\Sigma)$$

Značkou rovnice nemusí být pouze číslo. TEX čte vše po `\eqno` a `\leqno` v matematickém režimu, takže nepotřebujeme `$...$` kolem `\Sigma`.

Změna prokladu

Někdy je žádoucí změnit proklad (velikost mezery mezi řádky). Ve 3. kapitole jsme již použili globální příkaz `\openup`, který pracoval stejně v `\displaylines`, `\eqalign` nebo `\eqalignno` či `\leqalignno`. Musíme si zapamatovat, že `\openup` musí být použit před voláním makra, které sází zarovnané řádky, nikoli uvnitř něho: `\openup 4pt\displaylines{...}`.

Opakované volání `\openup` se sčítá. Pokud napíšeme `\openup2mm`, za kterým následuje `\openup3mm`, dostaneme stejný výsledek jako při `\openup5mm`. Můžeme také použít zápornou dimenzi, abychom proklad zmenšili.

S příkazy `\matrix`, `\pmatrix`, `\cases` a některými dalšími nebude `\openup` fungovat. První věc, kterou tato makra udělají je, že nastaví proklad na výchozí hodnotu. PlainTEX ukládá tuto hodnotu v dimenzi `\normalbaselineskip`, `\normallineskip` a `\normallineskiplimit`. Chceme-li řídit mezery v matici, musíme je změnit na `\baselineskip`, `\lineskip` a `\lineskiplimit`.

Oddělování dvou řádků

Pokud chceme oddělit pouze dva následující řádky, vyřešíme to vložením `\noalign{\vskip...}` mezi dva řádky:

```
$$\eqalign{
... & ... \cr
\noalign{\medskip}
...& ... \cr
}$$
```

Můžeme použít i záporný rozměr, např. `\noalign{\vskip-3pt}`, jestliže chceme řádky přiblížit k sobě.

Pomocí značky `\noalign{...}` můžeme mezi řádky vkládat nejen mezeru, ale jakýkoliv materiál, třeba i text [1].

Na závěr práce ukážeme sazbu jedné krásné formule – vzorce pro střední hodnotu binomického rozdělení:

```
$$P(X=k) = {n\choose k} p^k(1-p)^{n-k}$$
```

$$P(X=k) = \binom{n}{k} p^k(1-p)^{n-k}$$

```
$$\displaylines{
E(X)=\sum_{k=0}^n {n\choose k} p^k(1-p)^{n-k} =
\sum_{k=0}^n {n! \over (k-1)!(n-k)!}p^k(1-p)^{n-k}= \cr
np \sum_{k=1}^n {n-1 \choose k-1}p^{k-1}(1-p)^{n-k} = \cr
np \sum_{j=0}^{n-1} {n-1 \choose j} p^j (1-p)^{n-1-j} =
np[p+(1-p)]^{n-1}=np\cr
}$$
```

$$\begin{aligned}
E(X) = \sum_{k=0}^{n} \binom{n}{k} p^k(1-p)^{n-k} = \sum_{k=0}^{n} \frac{n!}{(k-1)!(n-k)!} p^k(1-p)^{n-k} = \\
= np \sum_{k=1}^{n} \binom{n-1}{k-1} p^{k-1}(1-p)^{n-k} = \\
= np \sum_{j=0}^{n-1} \binom{n-1}{j} p^j(1-p)^{n-1-j} = np[p+(1-p)]^{n-1} = np
\end{aligned}$$

Vidíme, že i takový vzorec lze vysázet poměrně jednoduše.

Závěr

Hlavním cílem bakalářské práce bylo přehledným a logickým způsobem zpracovat možnosti, které poskytuje plainTEX pro sazbu matematických formulí. Nejdříve jsem sázela jednoduché matematické formule, poté jsem se pokusila vytvořit i složitější víceřádkové formule.

Mým cílem bylo naučit se pracovat v systému plainTEX alespoň na základní úrovni. Formule vysázené plainTEXem lze totiž velmi snadno zařadit i do dokumentů, určených pro sazbu LATEXem, který se dnes používá nejčastěji. I já jsem postupovala tímto způsobem.

Pokud LATEX „odmítl“ formule vysadit, což například nastalo kromě makra `\displaylines` u víceřádkových formulí, řešením bylo použití knihovny plain, která přináší prostředí stejného jména. V prostředí plain již vše fungovalo správně.

Domnívám se, že ukázané možnosti sazby (i právě uvedené řešení možného problému se zařazením plainTEXovských formulí do LATEXu), pomůže všem, kteří z různého důvodu potřebují zařadit část matematického textu vysázeného plainTEXem do dokumentu LATEXu.

Literatura

[1] Seroul, R., Levy, S.: A Beginner's Book of TeX. Springer-Verlag, New York–Berlin–Heidelberg, 1991.

[2] Knuth, D. E.: The TeXbook. Addison-Wesley, Reading, 1991, 12ed.

[3] Olšák, P.: TeXbook naruby. Konvoj CSTug, Brno, 2001 [online] dostupné z: ftp://math.feld.cvut.cz/pub/olsak/tbn/tbn.pdf [cit. 1.4.2014].

[4] Olšák, P: TeX pro pragmatiky. [online] dostupné z: http://petr.olsak.net/ftp/olsak/tpp/tpp.pdf [cit. 1.4.2014].

[5] Doob, M: Jemný úvod do TeXu. CSTUG, Praha, 1993.

[6] Dokumentace obsažená v distribuci TeXu, složky `\texlive\2014\texmf-dist\doc`.