

**Jemný úvod do T<sub>E</sub>Xu**  
**Manuál pro samostatné studium**

Základní verzi napsal  
Michael Doob  
Department of Mathematics  
The University of Manitoba  
Winnipeg, Manitoba, Canada R3T 2N2

Volně přeložili a značně upravili  
Josef Daneš a Jiří Veselý  
s nezanedbatelnou pomocí Oldřicha Ulrycha, Karla Horáka a Petra Olšáka

Československé sdružení uživatelů T<sub>E</sub>Xu (C<sub>S</sub>TUG)

Praha 1993

# Úvod

---

Začneme špatnou zprávou:  $\TeX$  je rozsáhlý a komplikovaný program, který je určen k atraktivnímu tisku. Tato skutečná komplikace může způsobit občas neuvěřitelné věci. A nyní lepší zpráva: jednoduchý hladký text vysázíme pomocí  $\TeX$ u snadno. Proto lze začít se snadnými texty a postupovat ke komplikovanějším situacím.

Účelem tohoto manuálu je postupovat od samého začátku k mnohem složitějším věcem. Nepředpokládá se žádná předcházející znalost  $\TeX$ u. Postupem po sekcích podle časových možností se můžete naučit zvládat velmi různorodé texty.

Několik doporučení: v každé sekci jsou cvičení. Rozhodně je udělejte! Jediná cesta, jak se naučit  $\TeX$ , je používat ho. Ještě lépe je samostatně experimentovat; zkuste dělat variace na jednotlivá cvičení. Experimentováním v žádném případě nemůžete  $\TeX$  poškodit. Všimněte si na okrajích uvedených odkazů: vztahují se ke Knuthově knize *The  $\TeX$ book* – jsou značeny logem  $\TeX$ . Budete-li potřebovat o nějaké věci podrobnější informace, je to to správné místo, kde je hledat. Mimochodem, v tomto textu je pár drobných lží; jsou použity k zakrytí komplikací (chápu je jako básnické licence). Až budete zkušenější v zacházení s  $\TeX$ em, snadno je odhalíte.

$\TeX$  je veřejně přístupný program (public domain), který je k dispozici bez vydání za licenci. Byl vytvořen Donaldem Knuthem ze Stanfordovy univerzity na základě velkého projektu. Na trhu zaměřeném na zisk by tento program stál tisíce dolarů.  $\TeX$  Users Group (TUG) je nevýdělečná organizace, která distribuuje kopie  $\TeX$ u, inovuje programy a poskytuje informace o vývoji hardwaru a softwaru ve svých publikacích *TUGboat* a  *$\TeX$ niques*. Členský příspěvek není veliký: vezměte to, prosím, v úvahu.<sup>1</sup> Adresa je následující:

**$\TeX$  Users Group**  
**P.O. Box 869**  
**Santa Barbara, CA 93121–1041**  
**U.S.A.**

---

<sup>1</sup> V roce 1993 činil US \$ 60 a je v něm zahrnuto předplatné časopisu *TUGboat*.

Tento manuál by nevznikl bez pomoci jiných. Zejména opravy a návrhy od následujících lidí byly neocenitelnou pomocí: Robert Messer (Albion College), Anita Hoover (University of Delaware), John Lee (Northrop Corporation), Emily H. Moore (Grinnell College). [*... mnohem případnější text by měl být vložen na toto místo. Chtěl bych sem dát vaše jméno, pošlete mi proto své připomínky!*]

Několik lidí mi také poslalo část nebo celý manuál, který u nich používají. Několik jiných jsem obdržel jako reakci na tento text. Zejména různé typy skript od následujících lidí byly pro mne nejužitečnější: Elizabeth Barnhart (TV Guide), Stephan v. Bechtolsheim (Purdue University), Nelson H. F. Beebe (University of Utah) a Leslie Lamport (Western Digital Corporation), Marie McPartland-Conn a Luarie Mann (Stratus Computer), Robert Messer (Albion College), Noel Peterson (Library of Congress), Craig Platt (University of Manitoba), Alan Spragens (Stanford Linear Accelerator Center), Christina Thiele (Carleton University) a Daniel M. Zirin (California Institute of Technology).

**Poznámka překladatelů o autorovi:** Profesor Michael DOOB se narodil v roce 1942 ve Spojených Státech. Doktorát (Ph.D.) získal pod vedením Alana Hoffmana na City University v Novém Yorku v roce 1969. Je autorem několika desítek vědeckých prací a dvou knih o algebraické teorii grafů. Působí jako profesor na univerzitě v Manitobě a je  $\TeX$ editorem Kanadské matematické společnosti. Jeho žena se jmenuje Judita; má dvě děti, Lisu a Briana.

Knížka „Jemný úvod do  $\TeX$ u“ umožňuje čtenáři proniknout do tajů  $\TeX$ u s mimořádnou lehkostí a plně postačí pro takového čtenáře, který chce být obeznámeným expertem, ne však  $\TeX$ pertem. Autor poskytl překladatelům s mimořádnou ochotou dvě verze textu v elektronické podobě. Překlad byl pořízen ze starší verze s přihlédnutím k úpravám provedeným v novém textu. Autorovi, který dal svůj text k dispozici veřejnosti (public domain), patří díky celé stále rostoucí skupiny příznivců  $\TeX$ u u nás. Zde je též na místě upozornit čtenáře na to, že druhé vydání se od prvního dosti odlišuje.

Překladatelé děkují přátelům, kolegům, známým a studentům MFF UK, kteří jim obětavě pomohli s hledáním chyb při korekturách. Za zbývající chyby nesou odpovědnost pouze překladatelé. Protože věříme, že se text bude čtenářům líbit, doufáme, že vyjde patrně dříve či později znovu; prosíme proto o sdělení dalších nalezených chyb a nedopatření. Tato prosba je aktuální stále, protože vždy je co vylepšovat a počet začínajících uživatelů  $\TeX$ u patrně nějakou dobu ještě poroste. V dohledné době počítáme s vydáním podrobnější učebnice věnované makru `plain.tex`, přesto však věříme, že tato knížka bude čtenářskou veřejností přivítána jako potřebná pomůcka.

**Poznámka překladatelů k druhému vydání (kráceno):** Když jsme knížku Michaela Dooba překládali, netušili jsme, jaký vzbudí zájem. Náklad byl za krátkou dobu rozebrán a tak dochází k druhému vydání. Všimněme si krátce některých souvislostí.

Brzo po vytvoření  $\text{T}_{\text{E}}\text{X}$ u vznikla v USA mezinárodní organizace uživatelů  $\text{T}_{\text{E}}\text{X}$ u označovaná krátce TUG ( $\text{T}_{\text{E}}\text{X}$  Users Group), která již dvanáct let vydává časopis ryze věnovaný  $\text{T}_{\text{E}}\text{X}$ u. Rok 1989 znamená určitý zlom: po jistém váhání vytvořil Donald Knuth novou verzi  $\text{T}_{\text{E}}\text{X}$ u; je osmibitová (tj. dochází k dvojnásobnému zvětšení tabulek „použitelných“ písmen). Tím se  $\text{T}_{\text{E}}\text{X}$  3.x stal mnohem adaptovatelnější pro jazyky s větším počtem (akcentovaných) hlásek, tedy i pro češtinu. Potřeba národní standardizace vede ke vzniku stále nových odnoží TUGu, zpravidla na základě užívaného jazyka nebo podle geografických podmínek.

V Československu se pokusili uživatelé  $\text{T}_{\text{E}}\text{X}$ u sdružit pod hlavičkou Jednoty československých matematiků a fyziků. Po listopadu 1989 se podstatně zjednodušilo zakládání nových organizací: vzniklo Československé sdružení uživatelů  $\text{T}_{\text{E}}\text{X}$ u ( $\text{C}\mathcal{S}\text{T}_{\text{E}}\text{X}$ ). S využitím zahraničních souborů public domain softwaru,  $\text{S}\mathcal{B}\text{T}_{\text{E}}\text{X}$ u a později zejména  $\text{e}\text{T}_{\text{E}}\text{X}$ u, vytvořil Oldřich Ulrych z MÚ UK (MFF UK Praha) programový balík, který stručně označujeme  $\text{C}\mathcal{S}\text{T}_{\text{E}}\text{X}$ . Obsahuje prostředí a potřebné programové vybavení pro psaní textů v češtině, slovenštině a angličtině, snadno si jej však upravíte i pro psaní v jiných jazycích.

**Poznámka překladatelů k třetímu vydání:** Je nesporně příjemné psát předmluvu k třetímu vydání Doobovy knížky; máme pocit, že její přeložení bylo dobře vynaloženou prací. Knížka vychází na konci roku 1993, a tak je příležitost ke krátké bilanci.

Dnes již  $\text{T}_{\text{E}}\text{X}$  nepatří v České ani Slovenské republice k neznámým pojmům. Používá se ho k publikaci téměř všech u nás vydávaných matematických časopisů i řady časopisů jiného zaměření. Tiskne se jím ledacos, od složenek a školních časopisů až po časopisy o šachu, teologické publikace a desítky dalších věcí. Je nástrojem na tvorbu příspěvků pro zahraniční publikace, pro psaní diplomových i doktorských prací, pro vyhlášky, občasnky – zkrátka je téměř „děvečkou pro všechno“.

S rostoucí rolí  $\text{T}_{\text{E}}\text{X}$ u stoupá i význam Československého sdružení uživatelů  $\text{T}_{\text{E}}\text{X}$ u –  $\text{C}\mathcal{S}\text{T}_{\text{E}}\text{X}$ u (ano, stále se tak jmenujeme a snad ještě nějaký čas jmenovat budeme) a jisté standardizace užívaného vybavení. Nejde tolik o to, kdo a jaký bude používat editor – spíše o ovlivnění toho, aby různých kódování raději ubylo, než dále přibývalo. Je však jen příjemné, když zjistíte, že na všech pracovištích, na které u nás z toho či jiného důvodu zavítáte, se  $\text{T}_{\text{E}}\text{X}$  chová stejně; nemusíte se nikoho na nic ptát, víte o systému (skoro) všechno.

Dnešní podoba  $\text{C}\mathcal{S}\text{T}_{\text{E}}\text{X}$ u již není dílem jednotlivce. Pracovala na ní více či méně řada kolegů z  $\text{C}\mathcal{S}\text{T}_{\text{E}}\text{X}$ u. Těm dnes nesporně patří dík celé naší  $\text{T}_{\text{E}}\text{X}$ ovské uživatelské veřejnosti – ano, je totiž zcela přirozené, že je daleko více uživatelů  $\text{T}_{\text{E}}\text{X}$ u než členů  $\text{C}\mathcal{S}\text{T}_{\text{E}}\text{X}$ u.

Tvůrci  $\text{C}\mathcal{S}\text{T}_{\text{E}}\text{X}$ u však kromě důkazů a povzbuzení k další práci potřebují ještě něco navíc: zpětnou vazbu k uživatelům. Proto je důležité vědět, kdo v té skupině pracoval a co měl na starost. Přípomínky (nejlépe formou elektronické pošty) smě-

řijte proto přímo jednotlivým členům této skupiny. Důležité elektronické adresy naleznete mezi přílohami na konci této knížky.

Na tvorbě  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u se podíleli tito kolegové: Martin Bílý, správce jednoho z našich elektronických archivů dostupných po síti Internet (úprava  $\mathcal{L}\mathcal{A}\mathcal{T}\mathcal{E}\mathcal{X}$ ovských marker pro  $\mathcal{C}\mathcal{S}$ -fonty, testování, údržba instalace ve formě archívu), Miroslav Dont (stahování některých balíčků pro instalaci z veřejných archivů, testování), Karel Horák, současný předseda  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}$ u (závěrečná úprava zdrojových textů pro  $\mathcal{C}\mathcal{S}$ -fonty, kontakt s dodavateli programů šířených pouze v rámci  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}$ u), Janka Chlebíková (slovenské dělení slov a některé aspekty „poslovenšťování“), Ladislav Lhotka (české dělení slov, úprava  $\mathcal{C}\mathcal{S}$ -fontů, testování), Petr Novotný ( $\mathcal{C}\mathcal{S}\mathcal{B}\mathcal{I}\mathcal{B}\mathcal{T}\mathcal{E}\mathcal{X}$ ), Petr Olšák, autor instalačního programu (program MNU, konfigurace  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u, instalační a provozní dávky, podpůrná makra pro  $\mathcal{C}\mathcal{S}$ -fonty), Oldřich Ulrych, tvůrce původní instalace a koordinátor prací (koncepce stylu `czech.sty`, pomocné programy `cstocs`, `dviout`, zařazení  $\mathcal{C}\mathcal{S}$ -fontů do instalace podle skupin), Zdeněk Wagner ( $\mathcal{C}\mathcal{S}\mathcal{I}\mathcal{N}\mathcal{D}\mathcal{E}\mathcal{X}$ ) a ... – toto je alibistické místo, neboť i přes velkou snahu postihnout zásluhy na vzniku instalace jsme mohli někoho opomenout.

Na tomto místě bych se rád zmínil ještě o roli Petra Nováka, který dal k dispozici  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}$ u svoje původní  $\mathcal{C}\mathcal{S}$ -fonty a tak skupině ušetřil nezanedbatelné množství práce. Pavel Ševeček je autorem upravené verze  $\mathcal{C}\mathcal{S}\mathcal{E}\mathcal{D}$ u a Januš Drózd (s několika dalšími kolegy) je autorem programu  $\mathcal{T}\mathcal{E}\mathcal{X}\mathcal{S}\mathcal{P}\mathcal{E}\mathcal{L}\mathcal{L}$  pro kontrolu překlepů. Nebylo by vhodné opomenout roli Eberharda Mattese, jehož programy tvoří hlavní a nejdůležitější část instalace, i autorů několika dalších drobnějších programů (ovladač klávesnice, vlnka, ...).

Na závěr bychom rádi poděkovali všem, kteří se na práci  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}$ u podílejí a kteří umožňují vydávání informačního zpravodaje  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}$ u i dalších publikací o  $\mathcal{T}\mathcal{E}\mathcal{X}$ u v češtině a slovenštině. Ti všichni jsou sdruženi v organizaci

### **Československé sdružení uživatelů $\mathcal{T}\mathcal{E}\mathcal{X}$ u ( $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}$ )**

(viz stanovy na konci této publikace). Mezi jeho členy bychom vás rádi pozvali. Přihlášku a složenkou si můžete vyzvednout na adrese sdružení v MÚ UK, nebo v MÚ AV ČR na sekretariátě  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}$ u.

# Obsah

---

Úvod .....	i
Obsah .....	v
První kroky .....	1
Co je T <sub>E</sub> X a co T <sub>E</sub> X není .....	1
T <sub>E</sub> X se umí učit .....	3
O instalacích T <sub>E</sub> Xu, celkový pohled .....	5
Popis C <sub>S</sub> T <sub>E</sub> Xu a jeho instalace .....	9
Pracujeme s C <sub>S</sub> T <sub>E</sub> Xem .....	14
Čeština a slovenština v C <sub>S</sub> T <sub>E</sub> Xu .....	21
Jdeme na to! .....	25
T <sub>E</sub> X řídí všechno .....	27
Co T <sub>E</sub> X neudělá .....	28
Všechny velké a malé znaky .....	29
Všechny znaky jsou významné, ale některé jsou významnější .....	29
Sazba s akcentem .....	30
Tečky, pomlčky, uvozovky, .....	33
Různá písma .....	35
Úprava věcí příštích .....	38
Jednotky, jednotky, jednotky .....	38
Úprava stránky .....	39
Úprava odstavce .....	41
Různá písma .....	44
Poznámky pod čarou .....	45
V záhlaví a pod textem .....	46
Čeština a slovenština v C <sub>S</sub> T <sub>E</sub> Xu .....	47
{Skupiny, {skupiny, {a ještě skupiny}}}	50
Žádný strach z matematiky! .....	53
Mnoho nových symbolů .....	53
Zlomky .....	57
Indexy horní a dolní .....	57
Odmocniny, druhé mocniny a vůbec .....	58
Linky — nad i pod .....	58
Oddělovače velké a malé .....	59
Ach, ty speciální funkce .....	60
Na vědomost se dává! .....	61
Matice .....	62
Samostatné rovnice .....	63
Všechno v jednom řádku .....	65
Tab, tab, tabulka .....	65
Vodorovné zarovnání s rafinovaným vzorem .....	69
Dál válejte sami .....	74

Zkrátka a dobře .....	74
Vyplňování s parametry .....	76
Kterýmkoli jiným jménem .....	78
Chybovat je lidské .....	79
Zapomenuté konce .....	79
Překroucená a neznámá řídicí slova .....	79
Špatně pojmenované písmo .....	82
Popletená matematika .....	83
Nesprávné závorkování .....	84
Kutání trochu hlouběji .....	88
Velké soubory, malé soubory .....	88
Větší makra .....	89
Vodorovné a svislé linky .....	90
Rámečky v rámečku .....	91
Seznam řídicích znaků a slov .....	96
Řídicí znaky .....	96
Řídicí slova .....	96
Několik závěrečných informací .....	99
Vyřešeno s malou pomocí .....	105

# Kapitola 1

## První kroky

---

### 1.1 Co je $\TeX$ a co $\TeX$ není

$\TeX$  je program, který se používá k tisku pomocí počítačů; zejména je vhodný pro texty s mnoha matematickými symboly. Naučit se používat  $\TeX$  je tak trochu jako naučit se cizí jazyk. Nejprve se zdá, že se člověk musí naučit úplně ohromující řadu nových termínů, z nichž některé patří do typografie a jiné připomínají věci z programování. Stejně ale jako u cizích jazyků, začnete-li s jednoduchými myšlenkami a obraty, budete brzo znát dost na řešení většiny situací a komplikovanější tisk se naučíte zvládat příležitostně později. Nejúplnějším zdrojem informací o  $\TeX$ u je nepochybně knížka **The  $\TeX$ book** od Donalda E. Knutha<sup>1</sup>. Tato kniha je úplnou encyklopedií a slovníkem pro uživatele  $\TeX$ u a měl by ji mít k dispozici každý, kdo  $\TeX$  pravidelně používá. Tak, jako by však bylo téměř nemožné se naučit anglicky pouze čtením učebnice anglické gramatiky a slovníku, je obtížné se jednoduše naučit zacházet s  $\TeX$ em pouze čtením knihy **The  $\TeX$ book**.

Smyslem tohoto textu je seznámit čtenáře se základními myšlenkami  $\TeX$ u v tom rozsahu, aby jím mohl rutinně vytvářet texty. Po zvládnutí základů bude moci náročnější uživatel vytvářet rozmanité nebo původní věci pomocí Knuthovy knihy **The  $\TeX$ book**. V této části podáme přehled toho, čeho chceme dosáhnout. Vymezíme to, co  $\TeX$  dělá a co neumí. Zároveň se naučíme několik užitečných termínů a budeme umět dokonce již něco vytisknout.

Podívejme se nejprve na to, co musíme udělat, jestliže chceme vytvořit nějaký text pomocí  $\TeX$ u. Prvním krokem je vytvoření souboru, který se pak  $\TeX$ u předloží ke čtení. Tento soubor se obvykle nazývá *vstupní soubor* ( $\TeX$  file). Lze ho vytvořit pomocí jednoduchého textového editoru (ve skutečnosti, pokud používáte nestandardní textový procesor, musíte se ujistit, že váš text lze uložit v ASCII nebo ve tvaru bez jakýchkoli vlastních řídicích znaků). Program  $\TeX$  pak čte váš vstupní soubor a vytvoří soubor s extenzí *dvi* (*dvi* file). V tomto případě jde o jistou zkratku: **DVI** je odvozeno od **DeVice Independent**. Tento soubor není čitelný, přinejmenším pro lidi. Avšak soubor *dvi* se čte potom jiným programem, který nazýváme *ovladač* (*driver*, resp. *device driver*). Ten teprve vytvoří soubor, který je pro lidi (po vytištění) čitelný. Proč zvláštní soubor? Tentýž soubor *dvi* lze zpracovat různými ovladači a dostat tak výstup na jehličkové tiskárně, laserové tiskárně, na monitoru nebo na osvitové jednotce (*phototypesetter*). Jestliže jste vytvořili soubor *dvi*, který dává správný výstup například na obrazovce, můžete si být jisti, že dá přesně totéž (samozřejmě v kvalitnější podobě) na laserové tiskárně. To při

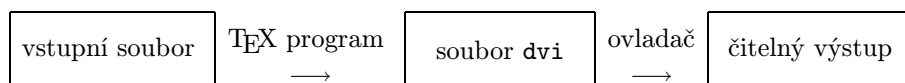
$\TeX$ book:  
23

---

<sup>1</sup> Addison-Wesley, Reading, Massachusetts, ISBN 0-201-13447-0, 1992, 22. vyd.



přípravě textů umožňuje odstranit velkou část chyb před prvním tištěním. Celý proces lze chápat jako postup podle schématu:



To ale znamená, že nevidíme výsledek naší práce v konečné podobě v době, kdy ho vytváříme pomocí klávesnice počítače nebo terminálu – musíme trochu počkat.

V případě  $\text{\TeX}$ u se trochu trpělivosti vyplatí, neboť můžeme při psaní užívat mnoho symbolů, jejichž psaní bychom normálně při použití většiny textových editorů či procesorů nezvládli. Navíc sazba je dělána s větší přesností a vstupní soubory lze přitom jednoduše posílat pomocí elektronické pošty mezi jednotlivými počítači nebo je přenášet na magnetickém médiu.

V této knize se hlavně soustředíme na první krok, tj. na vytvoření vstupního ( $\text{\TeX}$ ového) souboru. Existují dva způsoby zpracování tohoto souboru  $\text{\TeX}$ em: dávkový (tzv. batch mode) nebo interaktivní. Při dávkovém způsobu zpracování předložíte soubor počítači ke zpracování a na konci zpracovávání dostanete výsledek. V interaktivním způsobu se běh programu může přerušit pro přijetí dalších instrukcí od uživatele, tj. uživatel může být v interakci s programem. Interaktivní použití  $\text{\TeX}$ u umožňuje uživateli opravu některých chyb, v dávkovém módu  $\text{\TeX}$  opravuje chyby sám, jak nejlépe umí. Prvému způsobu se dává samozřejmě přednost. Všechny implementace  $\text{\TeX}$ u na osobních počítačích a většina implementací na velkých počítačích umožňují interaktivní zpracovávání, na některých velkých počítačích je ale možné jen zpracování dávkové.

[Poznámka překladatelů: Považovali jsme za vhodné následující text v odstavcích 1.2 až 1.6 **podstatně** rozšířit a přidat informace, o které je podle názoru našich kolegů největší zájem. Text zahrnuje tuto problematiku:

- Obecný přehled o instalacích  $\text{\TeX}$ u.
- Základní informace o tzv. formátech  $\text{\TeX}$ u, jako je plain  $\text{\TeX}$ ,  $\text{\LaTeX}$  apod.
- Co je třeba udělat, chceme-li na našem počítači pracovat s  $\text{\TeX}$ em (instalace).
- Jak se s  $\text{\TeX}$ em zachází.
- Co lze vyčíst ze souboru `log`.
- Jak si prohlédnout a/nebo vytisknout soubor `dvi`.

S ohledem na situaci v ČR a SR jsme se rozhodli i ve třetím vydání podrobně zmínit v této části zacházení s  $\text{\TeX}$ em na IBM PC-kompatibilním osobním počítači a popsat u nás nejrozšířenější instalaci zvanou  $\text{\CSTeX}$ . Jedině tato část je závislá na používaném počítači – nevztahuje se k ní žádný odkaz z jiné části textu. Obsah byl původně určen navrženou osnovou pocházející od M. Dooba. Texty ve zmíněných odstavcích 1.2 až 1.6 připravil pro toto vydání Petr Olšák.]

## 1.2 T<sub>E</sub>X se umí učit

Ze vstupního souboru čte T<sub>E</sub>X jednak vlastní text určený pro sazbu, jednak příkazy, kterými autor vyjadřuje své typografické požadavky (například typy písma, délky řádků apod.). Tyto příkazy (též řídicí slova, anglicky control sequences) jsou kombinovány s vlastním textem ve stejném vstupním souboru, přičemž výsledkem zpracování je soubor *dvi*, který obsahuje úplný popis vzhledu jednotlivých stránek – umístění jednotlivých písmen na stránce je zde definitivní. Ovladače pak tento popis převedou do čitelné podoby na obrazovku, na papír nebo jiné médium pro reprodukci.

T<sub>E</sub>X při svém běhu nečte pouze vstupní soubor s textem kombinovaným doprovodnými příkazy. Při startu nejprve přečte rozsáhlý soubor definic příkazů, které pak autor může ve svém textu použít. Před načtením tohoto souboru definic je T<sub>E</sub>X vybaven pouze zhruba třemi stovkami tzv. primitivních příkazů, které tvoří základní stavební kameny pro příkazy nové, „naučené“ z definic. Těmito primitivními příkazy je sice možné říci T<sub>E</sub>Xu jakýkoli požadavek, jsou však koncipovány jako příkazy „nízké úrovně“, které uživatel ve svém vstupním souboru většinou nepoužije. Uživatel tedy ve svém textu používá příkazy „vyšší úrovně“, které se T<sub>E</sub>X naučil při startu čtením ze souboru definic (též říkáme soubor *maker*). Tento soubor je textový soubor, ovšem v této podobě se T<sub>E</sub>Xu v běžném provozu nepředkládá. T<sub>E</sub>X je totiž také schopen načíst makra a „naučit se je“ a pak obraz vytvořených struktur v paměti uložit do binárního souboru. Tomuto procesu se říká *inicializace* daného souboru maker. V běžném provozu pak T<sub>E</sub>X při svém startu čte tento binární soubor, což je nesrovnatelně rychlejší „učení“. Binárnímu souboru, který T<sub>E</sub>X načítá při svém startu (a který byl někdy inicializován z nějakého textového souboru maker) obvykle říkáme *formát*.

Ve většině implementací je schopnost programu T<sub>E</sub>X „učit se“ dotažena do důsledku. Při inicializaci je totiž T<sub>E</sub>X schopen vytvořenou binární strukturu formátu zahrnout do nového spustitelného kódu a tak sestavit nový program T<sub>E</sub>X, který už toho umí podstatně více než jeho předchůdce. Tento poněkud rozdílný technický přístup nemá pro uživatele žádný podstatnější význam; z logického hlediska je totiž jedno, zda se spustí výchozí (virgin – panenský) T<sub>E</sub>X, jemuž se předloží při startu příslušné soubory maker ke čtení, nebo se použije nově vytvořený a „chytřejší“ spustitelný kód. Na počítačích typu PC se tvorba nových spustitelných kódů nedá realizovat; lze ale vytvářet zvláštní binární soubory s formáty.

V každé instalaci T<sub>E</sub>Xu najdeme několik mezinárodně uznávaných formátů. Základem je formát vyvinutý samotným autorem T<sub>E</sub>Xu. Formát se jmenuje *plain* (nebo *plain T<sub>E</sub>X*). Tento formát je zcela popsán (včetně důkladného rozboru jeho souboru definic) v knize **The T<sub>E</sub>Xbook**. Výhodou tohoto formátu je jeho jednoduchost a stabilita (nebude dále měněn). Na druhé straně, pokud autor požaduje řešit složitější typografické úkoly, např. automatické vytváření obsahu, křížových odkazů apod., musí je formulovat většinou velice komplikovaně pomocí příkazů „nízké úrovně“.

Druhým velmi často používaným formátem je  $\text{\LaTeX}$ . Tento formát vytvořil Leslie Lamport a uživatelský manuál, který k tomuto formátu napsal, se jmenuje  **$\text{\LaTeX}$  – A Document Preparation System**. Zde najdete už mocné nástroje pro vytváření křížových odkazů, obsahu, rejstříku, živých záhlaví a třeba i pro tvorbu jednoduchých obrázků. Mimoto je tento formát založen na důsledném dodržování logické struktury dokumentu (členění na kapitoly, odstavce, pododstavce apod.) a na používání povinných a doplňkových souborů maker, tzv. *stylů*. Tyto soubory se rovněž načítají  $\text{\TeX}$ em při jeho startu (většinou hned za formátem), přičemž v těchto makrech se upřesňuje způsob zpracování dokumentu (zda to bude kniha nebo dopis) a případně se dodefinovávají další uživatelsky zajímavé příkazy. Výhoda tohoto přístupu je zřejmá. Odborník na typografii a  $\text{\TeX}$  připraví styl a dokumentuje jej. Uživatel pak použije příkazy podle dokumentace a přitom nemusí pořádně umět ani  $\text{\TeX}$ , ani typografické zásady.<sup>1</sup>

Dalším hojně používaným formátem (převážně v matematických kruzích) je  $\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\text{\TeX}$ . Formát vytvořil Michael Spivak a dokumentoval jej v knize **The Joy of  $\text{\TeX}$** . Pomocí tohoto formátu lze poměrně snadno psát i složité matematické texty. Formát  $\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\text{\TeX}$  a zejména pak stylový soubor `amsppt.sty` byly vytvořeny pro potřeby Americké Matematické Společnosti (AMS), která přijímá články psané v tomto formátu. Tento formát není vybaven příkazy pro automatickou tvorbu obsahu a křížových referencí. Na druhé straně definuje přístup k velkému množství nejrůznějších nestandardních písem, použitelných především v matematických formulích.

Vylepšená verze formátu  $\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\text{\TeX}$  má jméno  $\text{L}\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\text{\TeX}$ . Tento formát (rovněž od M. Spivaka) už nabízí příkazy pro generování obsahu a křížových referencí, ovšem není příliš hojně užívaný. Instalační balík tohoto formátu obsahuje zajímavý soubor maker pro sazbu tabulek a ještě zajímavější program `DVIPASTE`, který umožňuje vytvářet  $\text{\TeX}$ em tabulky a složité diagramy zvlášť mimo hlavní dokument. Program totiž dokáže „spojit“ soubor `dvi` obsahující hlavní dokument bez tabulek se souborem `dvi`, v němž je výsledek sazby tabulek. Tak vznikne soubor `dvi` obsahující obojí (přitom tabulky jsou tam, kde si přejeme).

Další vývoj formátů se zaměřil na vytvoření souboru maker, které uživateli umožní velice snadno zavést a použít v podstatě jakýkoli řez písma (tzv. *font*) dostupný v dané instalaci. Přitom tato makra ctí členění řezů písma podle typografických zvyklostí. Tak vznikl soubor maker známý jako `NFSS` (New Font Selection Scheme). Tento soubor se dá kombinovat s libovolným formátem (tedy například i s `plain \TeX`em), ovšem nejběžnější je jeho kombinace s  $\text{\LaTeX}$ em. Balík maker, obsahující  $\text{\LaTeX}$ , `NFSS` a některé  $\text{\LaTeX}$ ovské styly, které umožní, aby se  $\text{\LaTeX}$  choval skoro jako  $\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\text{\TeX}$ , se nazývá  $\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\text{\LaTeX}$ .

Na balíky maker můžeme pohlížet jako na další software, který je napsaný v jazyce  $\text{\TeX}$ u a používaný  $\text{\TeX}$ em. Pro zajímavost např.  $\text{\LaTeX}$  je soubor definic,

---

<sup>1</sup> Myšlenka maker a stylů je velmi podobně použita i v systémech `SGML`; zde jsou „styly“ označovány zkratkou `DTD`.

obsahující zhruba 10 000 řádků napsaných v tomto jazyce. Všechny zde zmíněné balíky jsou volně šířené stejně jako samotný program  $\text{T}_{\text{E}}\text{X}$ . V této příručce se budeme zabývat *pouze* psaním textů pod formátem plain. Tyto texty jsou použitelné téměř beze změn i v  $\mathcal{A}\mathcal{M}\mathcal{S}\text{-T}_{\text{E}}\text{X}$  (existuje zde přenositelnost z „nižšího“ formátu), ovšem bohužel  $\text{L}\text{A}\text{T}_{\text{E}}\text{X}$  není s formátem plain slučitelný. Chcete-li se zaměřit pouze na psaní v  $\text{L}\text{A}\text{T}_{\text{E}}\text{X}$ u, budete se muset poohlédnout po jiné příručce. Na druhé straně dobrá znalost plain  $\text{T}_{\text{E}}\text{X}$ u vám umožní orientovat se velice rychle v dalších formátech, porozumět souborům maker a dokonce některá jednoduchá makra si samostatně napsat.

### 1.3 O instalacích $\text{T}_{\text{E}}\text{X}$ u, celkový pohled

$\text{T}_{\text{E}}\text{X}$  samotný je vlastně jakýsi „automatický sazeč“. Je vybaven algoritmy na zpracování textu, jako je například algoritmus, který rozhoduje o rozdělení textu v odstavci do řádků, jiný algoritmus umožňuje sazbu tabulek nebo libovolně komplikovaných matematických vzorců apod. Také, jak jsme uvedli v předchozím odstavci, je  $\text{T}_{\text{E}}\text{X}$  vybaven schopností „učit se“ nové příkazy. Všechny algoritmy jsou navrženy tak, aby pokryly veškeré typografické požadavky, které se mohou vyskytnout při práci s textem. Navíc tyto algoritmy jsou zcela *nezávislé* na použitém počítači. Proto je  $\text{T}_{\text{E}}\text{X}$  dodáván v podobě zdrojového textu, aby jej šlo použít na jakémkoli počítači vybaveném příslušným překladačem. Tímto překladačem je překladač jazyka Pascal, přičemž se nepožaduje žádné rozšíření, které není definováno v normě jazyka. Pro instalace na systémech UNIX s překladačem jazyka C byl vyvinut program, který zdrojový text v Pascalu transformuje do C. Tím je  $\text{T}_{\text{E}}\text{X}$ u otevřena cesta k dalším (novějším) systémům.

Na tomto místě je nutno poznamenat, že výše uvedený postup získání spustitelného programu  $\text{T}_{\text{E}}\text{X}$  ze zdrojového textu klade poměrně vysoké nároky na paměťové možnosti počítače. Proto se tento postup nedá použít na „malých“ počítačích, jakými jsou například počítače typu PC. Zde byl zdrojový text upraven a místy přepsán do strojového jazyka, aby šlo program na těchto počítačích použít. Zdrojové texty nejsou obvykle veřejně šířeny. Místo nich je šířen pouze spustitelný kód programu jako součást ucelených instalací. Každý takto sestavený program musí projít testovacím programem vytvořeným D. Knuthem, aby mohl užívat ve jménu označení  $\text{T}_{\text{E}}\text{X}$ . Tím je zaručena téměř absolutní kompatibilita všech instalací  $\text{T}_{\text{E}}\text{X}$ u.

Protože program  $\text{T}_{\text{E}}\text{X}$  se zaměřuje na tu část typografického zpracování textu, která není závislá na použitém počítači a výstupním zařízení, samotný by nám moc užítku nepřinesl. Potřebujeme tedy již zmíněné programy – ovladače, které výstup z  $\text{T}_{\text{E}}\text{X}$ u (soubor `dvi`) převádějí do čitelné podoby. Tyto programy jsou samozřejmě závislé na technických možnostech obrazovky nebo tiskárny. Taky potřebujeme nějaký editor, kterým připravujeme vstupní soubory pro  $\text{T}_{\text{E}}\text{X}$ . I editor bývá závislý na použitém výpočetním prostředí. Teprve tyto programy vytvoří možnost skutečně vysázet  $\text{T}_{\text{E}}\text{X}$ em dokument tak, jak potřebujeme.

Každý uživatel používá pro přípravu vstupních textů svůj oblíbený editor, tj. ten, na který je zvyklý a kterému dává přednost před ostatními. Začínáme-li nebo máme-li možnost si vybrat, uijeme raději jednoduchý rychlý editor s možností práce s více soubory najednou. Důležité přitom je, aby výsledný soubor byl v ASCII kódu. Editor si nesmí do souboru přihrávat další vlastní řídicí znaky, které by kolidovaly s  $\text{\TeX}$ ovým použitím a nám komplikovaly zpracování. Také je podstatné, zda je editor v téže podobě dostupný na více systémech (např. editor Emacs). Jsme-li však zvyklí na nějaký editor nebo textový procesor, který je z tohoto hlediska nevhodný, nebývá obtížné výstupní soubor eventuálně lehce upravit; existují programy, které konvertují výstup z populárních textových procesorů a umožňují tak pohodlnější přípravu vstupního souboru (existuje například preprocesor pro  $\text{\TeX}$  z WordPerfectu).

V konkrétních instalacích  $\text{\TeX}$ u tedy najdeme nejen program  $\text{\TeX}$  samotný, ale i spoustu doprovodných programů (ovladače `dvi` souborů, editor apod.), a také tam jsou fonty (řezy písma), které jsou digitalizovány do rozlišení, které odpovídá použitým připojeným zařízením. Fonty používané v  $\text{\TeX}$ u pocházejí (není to ovšem pravidlem) z rodiny Computer Modern, kterou vytvořil autor  $\text{\TeX}$ u. Vyšel přitom z písem, která byla vyvinuta už dávno před existencí počítačů. Písmo je definováno „vektorově“ (tedy nezávisle na rozlišení výstupního zařízení) v souborech, které jsou určeny ke zpracování programem METAFONT. Tento program (byl rovněž vyvinut autorem  $\text{\TeX}$ u a je vybaven podobnými silnými nástroji jako  $\text{\TeX}$ ) čte uvedené soubory a vytváří bitové mapy písmen. Tyto bitové mapy pak používají ovladače souborů `dvi`.

Poznamenejme ještě, že  $\text{\TeX}$  dokáže pracovat s *jakýmkoli* fonty, ke kterým se dají získat metrické údaje. Fyzicky se může jednat například o fonty PostScriptové či jiné, implementované například jen v některých sázecích systémech. Proto by vás nemělo překvapit, že  $\text{\TeX}$  sází texty v podstatě ve všech světových jazycích včetně japonštiny, hebrejštiny, arabštiny apod. V takových případech může být užitečný mechanismus tzv. virtuálních fontů, který autor  $\text{\TeX}$ u definoval teprve nedávno (r. 1990).

V instalacích  $\text{\TeX}$ u se vyskytují také další „doplňkové“ programy. Uvedme jenom některé. Program MAKEINDEX umožňuje sestavovat v  $\text{\TeX}$ u rejstříky, přičemž je dbáno na mnoho pravidel uspořádání podle abecedy. Tento program byl pro naše potřeby rozšířen a vznikl program  $\mathcal{C}$ SINDEX, který třídí podle normy českého a slovenského jazyka. Také existuje program Bi $\text{\TeX}$ , který umožní pracovat v  $\text{\TeX}$ u s databází publikací, z níž se některé položky podle výběru autora a explicitních citací automaticky umístí do závěrečných referencí literatury ve tvaru, který je v odborných člancích běžný. Pro český a slovenský jazyk vznikla verze zvaná  $\mathcal{C}$ SBi $\text{\TeX}$ . Také bývá velmi užitečný korektor překlepů, který zkontroluje vstupní text pro  $\text{\TeX}$  po jazykové stránce.

I když je v současné době  $\text{\TeX}$  implementován také na jiných počítačích, většina jeho uživatelů v ČR a SR bude pracovat ještě nějakou dobu na osobních počítačích pod systémem MS DOS. Veřejně přístupných implementací  $\text{\TeX}$ u pro

uvažované osobní počítače je již dost; jmenujme alespoň některé:  $\text{DosTeX}$  (autorem je Gary Beihl),  $\text{SBTeX}$  (autor Wayne G. Sullivan) a  $\text{emTeX}$  (tento je srovnatelný s komerčními implementacemi a jeho autorem je Eberhard Mattes); zdaleka však výčet nekončí, velmi zdařilou implementací je např.  $\text{PubliCTeX}$ . Poslední dva programové balíky jsou šířeny v rámci skupiny německy píšících uživatelů  $\text{TeXu}$ , která má jméno DANTE a se kterou naše sdružení  $\text{CSSTUG}$  úzce spolupracuje.

Po zpracování dokumentu  $\text{TeXem}$  si můžeme získaný soubor  $\text{dvi}$  prohlédnout na obrazovce, eventuálně ho přímo vytisknout. To druhé však v našich podmínkách děláme zpravidla až po několika pokusech provést všechny opravy (překlepy v textu atp.). Na osobních počítačích s dobrou grafikou lze získat poměrně přesnou představu o výsledném dokumentu prohlédnutím na obrazovce. Zatím je samozřejmě podstatný rozdíl mezi rozlišovací schopností obrazovek a tiskáren – proto systémy typu WYSIWYG (What You See Is What You Get, tedy „co vidíš, to dostaneš“) jsou spíše WYSIAWYG (kde „A“ je od Almost neboli „skoro“).  $\text{TeX}$  na osobních počítačích není (až na jisté výjimky) tohoto typu a umožňuje optickou kontrolu výstupu pomocí obrazovky monitoru jen s pomocí ovladače souboru  $\text{dvi}$ , tedy až po zpracování alespoň části vytvářeného dokumentu.

Uvedený způsob není sice příliš operativní, ale pro většinu uživatelů  $\text{TeXu}$  je programátorská filosofie typu „co je psáno, to je dáno“ neocenitelnou předností. Tento dávkový přístup typografického zpracování (tj. vstupní textový soubor s vlastním textem a popisem typografických požadavků je zpracován do výstupního souboru  $\text{dvi}$ ) umožňuje řešit úlohy, které jsou WYSIWYG systémům nedosažitelné. Například vstupem do  $\text{TeXu}$  mohou být (po načtení příslušných maker) různé databáze a výstupem jsou třeba typograficky perfektně upravené slovníky, jízdní rády nebo třeba jen různé formuláře. Také je třeba si uvědomit, že interaktivní nabídky WYSIWYG systémů vymezují pro uživatele konečné mantinely a neumožňují jim absolutně přesně formulovat jakýkoli typografický požadavek.

V současné době vznikají i implementace, umožňující vytváření vstupního textu téměř ve WYSIWYG formě. Používají se zde editory s grafickým rozhraním a možností řešení většiny typografických úkolů rovnou z interaktivní nabídky, přičemž na obrazovce *zhruba* vidíme, jak bude dokument vypadat po zpracování  $\text{TeXem}$ . Funkce „save“ ukládá přitom dokument do textové podoby připravené pro vstup ke zpracování  $\text{TeXem}$ . Lidem, kteří píší v  $\text{TeXu}$  častěji a využívají jej „naplno“, tyto systémy příliš svazují ruce. Tento směr vývoje podpory  $\text{TeXu}$  ale je pro mnohé velmi užitečný. Vždyť v  $\text{TeXu}$  píší nejen přímo experti, ale i sekretářky, písařky apod., a pro ně to je vítaná pomůcka. Je to podstatně lepší přístup, než jaký nabízejí drahé a uzavřené typografické systémy, které nemají s  $\text{TeXem}$  nic společného. V případě použití zmíněného editoru totiž vždy k výslednému (uloženému) textu v podobě  $\text{TeXovského}$  vstupu může přistoupit odborník a může dodatečně udělat libovolné úpravy. Bohužel v balíku  $\text{CSTeXu}$  takový editor nenajdeme, protože za programy tohoto typu je nutné zatím zaplatit dosti velké částky.

S výjimkou uvedeného případu je dnes již vcelku zbytečné používat komerčně šířených implementací  $\text{TeXu}$ , alespoň v oblasti práce na PC. Mohou však nastat

důvody, pro něž to je výhodné (kompatibilita *speciálních* příkazů při spolupráci se zahraničním partnerem apod.). Pro zajímavost jim věnujme několik slov: skrývají se za označeními jako např. PCT<sub>E</sub>X (Personal T<sub>E</sub>X Inc.), TurboT<sub>E</sub>X (Kinch Computer Company) a CT<sub>E</sub>X (Micro Publishing Systems, Inc.); jde o různě obsažené soubory programů, které se liší jak cenou, tak i poskytovanými výhodami, pohodlím, rychlostí atp.

Jak již bylo řečeno, k prohlédnutí jisté aproximace výsledného tisku je potřeba ovladač souboru *dvi* pro obrazovku (previewer). Komerční produkty jsou v této oblasti někdy dokonalejší. Nejznámějšími jsou Preview a MaxView, resp. softJET (ten ale pracuje se souborem *hp* pro laserovou tiskárnu HP). Různé jejich přednosti a nevýhody nebudeme popisovat, poznamenáváme pouze, že tyto ovladače se zpravidla kupují zvlášť a jejich cena je srovnatelná s cenou jiných kompletních publikačních systémů. Připojme však doporučení: je-li to možné, spokojte se s tím, co nabízí *CS*T<sub>E</sub>X, který obsahuje vesměs volně šířené programy. Využívání těchto programů je v našich podmínkách velmi efektivní a dostačující.

Komerční trh se v poslední době orientoval podstatně jiným směrem. Nabízí uzavřené typografické systémy, které jsou vázané na konkrétní typ počítače a další hardwarové prostředky (scannery, tiskárny apod.). Zákazník zaplatí vysokou sumu za kompletní a definitivní „typografické studio“, v jehož středu je program na zpracování textů, který většinou nemá s T<sub>E</sub>Xem nic společného. Tento program nabízí konečné množství možností, které jsou jednoznačně ohraničeny interaktivní nabídkou. Celkem bez potíží lze vytvořit výstup, který je po typografické stránce špatný. To je zcela jiný přístup, než používá T<sub>E</sub>X. Až program T<sub>E</sub>X hlouběji poznáte, zjistíte, že to je typografický program neomezených možností, geniálně navržený a pracující s absolutní přesností. Jeho principy vycházejí ze stovky let starých tradic kultury písma a typografie. Také je to stále mezi programy a *nehrozí* žádné přechody na nové verze. Kompatibilita mezi všemi existujícími typy počítačů je také zaručena. Program za patnáct let své existence samozřejmě vyrostl z dětských nemocí. Všechny tyto přednosti jsou komerčním typografickým systémům naprosto cizí a nedosažitelné.

Výsledný vzhled dokumentu je podstatně závislý na použité tiskárně. Pro T<sub>E</sub>X se snažíme mít k dispozici co nejkvalitnější tiskárnu, neboť teprve v kombinaci s ní můžeme ocenit jeho dokonalost. Velmi hrubé výsledky lze získat na jehličkových tiskárnách typu LQ, lepší (a v našich podmínkách víceméně dokonalé) je použití laserové tiskárny. Za podstatnou považujeme skutečnost, že jsou dnes dostupné laserové tiskárny s rozlišením nejen 300 dpi (bodů na palec – dot per inch), ale i 600 dpi a více, které v mnoha případech postačí k vytvoření *velmi kvalitní* předlohy pro tisk příruček, skript a knih. Nejlepší řešení je nechat výsledný *dvi* soubor nasvítit na osvitové jednotce, což je zařízení s rozlišením několika tisíc dpi a s výstupem rovnou na film, který se stává předlohou pro kontaktní svícení ofsetové tiskové formy. Takové zařízení ale je příliš drahé a chceme-li využít služeb nabízených firmami, které osvitovou jednotku vlastní, musíme nejprve METAFONTEM digitalizovat fonty do příslušného rozlišení a pak ještě použít konvertor z formátu *dvi* do PostScriptu, což je jazyk, se kterým se lze s osvitovou jednotkou „domlu-

vit“. Bohužel neznáme (zatím) jedinou tuzemskou firmu nabízející osvit, která by tuto přípravu udělala sama a přijímala od zákazníka přímo dva soubory.

Nemáte-li tiskárnu k dispozici a připravujete-li dokument pouze s cílem získat jej v elektronické formě, je užití  $\text{\TeX}$ u do jisté míry málo efektivní.

Potřebujete-li získat další informace, např. o implementacích pro jiné počítače, sledujte elektronickou konferenci o  $\text{\TeX}$ u v ČR a SR. Stačí mít přístup k elektronické poště a přihlásit se do elektronické diskuse zasláním dopisu, který obsahuje jediný řádek

```
subscribe cstex
```

na adresu `listserv@cs.felk.cvut.cz`. Od té chvíle vám budou docházet všechny příspěvky směřované na adresu konference. Jsou to otázky a odpovědi, týkající se vesměs problematiky české a slovenské implementace  $\text{\TeX}$ u. Pokud byste chtěli sami poslat příspěvek do konference, musíte znát také její adresu a ta je: `cstex@cs.felk.cvut.cz`.

## 1.4 Popis $\mathcal{C}\mathcal{S}\text{\TeX}$ u a jeho instalace

Následující tři sekce obsahují velmi konkrétní informace o implementaci  $\text{\TeX}$ u, která je v současné době u nás nejdostupnější. Jedná se o balík  $\mathcal{C}\mathcal{S}\text{\TeX}$ , určený pro IBM PC kompatibilní počítače pracující pod systémem MS DOS. Tento balík je sestaven především z programů balíku `em $\text{\TeX}$`  a je vybaven širokou podporou práce s českým a slovenským jazykem. V současné době lze  $\mathcal{C}\mathcal{S}\text{\TeX}$  získat prakticky bezplatně prostřednictvím  $\mathcal{C}\mathcal{S}\text{TUG}$ u. I když informace o tomto balíku opět brzo zestárnou, nemělo by to být na škodu. Doufáme, že změny, k nimž dojde, budou vždy pouze k lepšímu.

Předpokládáme, že čtenář bude mít možnost si některé věci přímo na počítači vyzkoušet. Je nutné, aby to byl počítač vybavený pevným diskem a slušnou grafikou. Lepší rozlišení podstatně usnadňuje prohlížení výsledného textu na obrazovce, takže doporučujeme aspoň HGC grafiku (Hercules) nebo EGA, VGA, SVGA. . . Pro práci s  $\text{\TeX}$ em (při prohlížení jeho výstupu na obrazovce) je podstatnější rozlišovací schopnost karty (monitoru) nežli možnost pracovat s barvou.  $\text{\TeX}$  se spokojí s počítačem XT, v zásadě si však s koupí lepšího (rychlejšího) počítače kupujete možnost pracovat s  $\text{\TeX}$ em lépe, tj. s většími soubory apod. Na procesorech INTEL 80386 a lepších je možné spustit verzi tzv. Big- $\text{\TeX}$ u, která dokáže pracovat s rozsáhlými formáty s množstvím fontů a maker, jako je např.  $\mathcal{A}\mathcal{M}\mathcal{S}\text{-L}\mathcal{A}\text{\TeX}$ . Dnes je dostupnost vhodného vybavení otázkou finanční. Snad je však přeci jen vhodné toto upozornění: uvažte, že modernější počítač zestárne později a ušetří vám oči i čas.

Programový balík s  $\text{\TeX}$ em obsahuje většinou instalační program, který postupně vytvoří na pevném disku adresář pro celý systém s podadresáři, ve kterých se po instalaci zpravidla nakupí spousta programů. U námi popisované verze  $\mathcal{C}\mathcal{S}\text{\TeX}$



je to hlavní adresář `\emtex`, který se ale může jmenovat i jinak a může být umístěn i hlouběji ve stromové struktuře disku. To záleží na volbě v nabídce instalačního programu. V hlavním adresáři jsou umístěny všechny spustitelné programy. Dále se v tomto adresáři vytvoří podadresáře `\doc` s dokumentací, `\start`<sup>1</sup> se spouštěcími dávkami, `\cfg` s konfiguračními soubory a dávkami, `\texfmts` s formáty `TeXu`, `\texinput` s textovými vstupy pro `TeX`, `\tfm` s metrikami fontů, `\fonts` s bitovými mapami fontů a případně další.

K současné instalaci `CSTeXu` (popisujeme stav k 30. 10. 1993, jako každý soubor programů se však i `CSTeX` stále vyvíjí) připojme ještě několik poznámek: je založena na `TeXu` verze 3.14 a byla sestavena v MÚ UK z veřejně dostupného softwaru, ze softwaru vytvořeného v MÚ UK a softwaru získaného od různých autorů s jejich laskavým svolením. Je šířena Československým sdružením uživatelů `TeXu`. Cílem je poskytnout zájemcům legální cestou zdarma (resp. za manipulační poplatek) základní software, který by umožňoval používat `TeX` pro běžnou potřebu v českém, slovenském a anglickém jazyce.

Instalace `CSTeXu` v současné době zahrnuje:

- Osmibitové `CS`-fonty. Jde o rozšíření Knuthových Computer Modern fontů o některé znaky s kódy nad 128. Jedná se vesměs o akcentovaná písmena používaná v české a slovenské sazbě. Akcenty tvoří s písmenem nedílný optický celek. Původní metoda sestavování akcentovaných písmen „ze dvou dílů“ příkazem `\accent` (viz odstavec 2.2 v této knize) neposkytuje tak dobré výsledky jako při přímém použití `CS`-fontů.
- Kódování vstupních textů může být libovolné. Instalace podporuje kódování podle Kamenických, PC Latin 2 a KOI 8.
- Kromě standardních formátů (plain, `LATEX`, `AMS-TeX`) zde najdete formát `AMS-LATEX`, což je vlastně `LATEX` s použitím NFSS (new fonts selection scheme), dále `LAMSTEX` (s ním souvisí program DVIPASTE a různá makra pro tabulky a komutativní diagramy) a `SliTeX` (k vytváření folií (slides) pro zpětnou projekci).
- Je zahrnuto uživatelské menu, založené na programu MNU a na vzájemném volání různých dávek. Je použitelné i pro provoz na sítích.
- Je zahrnut editor `CSED`, zakoupený ve zjednodušené verzi pro členy `CSTUGu` od Pavla Ševečka.
- Slovníky pro `TeXSPELL` jsou nyní: anglický, český, slovenský (pouze pro členy `CSTUGu`).
- Je zahrnut `CSINDEX` na vytváření rejstříků podle anglických i našich národních pravidel.
- Najdete zde program `CSBiBTeX` pro evidenci literatury (zahrnuta jsou opět i naše národní pravidla).

---

<sup>1</sup> Tento adresář se zatím ve stávající verzi jmenuje `\bats`, ovšem od ledna r. 1994 připravujeme pro něj příznačnější jméno `\start`, které vystihuje, že je nutné se s ním zabývat při startu (instalaci a inicializaci) systému a že obsahuje startovací dávky.

- Jsou použity aktuální verze programů em $\TeX$ u z dílny Eberharda Mattese.
- Je přibalen zdrojový text překladu „ $\LaTeX$ – Stručný popis“.
- Možná zde najdete i některá další příjemná překvapení.

$\TeX$ nické požadavky na počítač, na němž chcete instalovat  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ , jsou:

- Počítač IBM PC compatible, procesor stačí 8086.
- Min. 512 KB RAM.
- Pevný disk s min. 6 MB volné paměti.
- Disketová jednotka HD ( $5\frac{1}{4}$  in nebo  $3\frac{1}{2}$  in) – nutná pouze pro instalaci z disket.
- Grafická karta v podstatě libovolná (při CGA je ale prohlížení výsledku velmi omezené).
- Tiskárna v podstatě jakákoli. Instalace podporuje (vybavením fonty) jehličkové tiskárny s rozlišením 240 dpi (9 jehel), 180 nebo 360 dpi (24 jehel) a 300 dpi (laserové tiskárny). Pro posledně jmenovanou tiskárnu stačí vybavení tzv. jazykem „hp“ ; PostScript není v tiskárně potřeba a jakékoli fonty vestavěné do tiskárny jsou z hlediska  $\TeX$ u zbytečné. Pro jiná výstupní zařízení jsou do instalace zahrnuty prostředky na možné vygenerování fontů (METAFONT, MFJOB).
- MS DOS ver. 3.30 nebo vyšší. Řídící dávky nepoběží v DOSu s nižší verzí kvůli příkazu call, nepoběží ani v NDOSu, kvůli odlišné interpretaci znaků ‘ a ^ . DR DOS nebyl testován.
- Pokud chcete používat tzv. Big- $\TeX$  (**tex386**), pak je třeba mít procesor 80386 nebo vyšší. Koprocesor není potřeba.

Distribuce obsahuje 18 disket. Jejich význam je následující:

1. Instalační program (pro všechny ostatní diskety) + základní software.
2. Základní software.
3. Rozšiřující software ( $\mathcal{A}\mathcal{M}\mathcal{S}\text{-}\mathcal{L}\mathcal{A}\mathcal{T}\mathcal{E}\mathcal{X}$ ,  $\mathcal{L}\mathcal{A}\mathcal{M}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ ,  $\mathcal{C}\mathcal{S}\mathcal{I}\mathcal{N}\mathcal{D}\mathcal{E}\mathcal{X}$ ,  $\mathcal{C}\mathcal{S}\mathcal{B}\mathcal{I}\mathcal{B}\mathcal{T}\mathcal{E}\mathcal{X}$ , ...).
4. METAFONT a zdrojové texty k fontům.
5. Disketa pouze pro členy  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}\mathcal{U}$  ( $\mathcal{C}\mathcal{S}\mathcal{E}\mathcal{d}$ ,  $\mathcal{S}\mathcal{P}\mathcal{E}\mathcal{L}\mathcal{L}$ ).
- 6–8. Fonty 180 dpi. Z toho diskety **6**, **7** – základní fonty
- 9–12. Fonty 240 dpi. Z toho diskety **9**, **10** – základní fonty
- 13–18. Fonty 300 dpi. Z toho diskety **13**–**15** – základní fonty.

Obsah jednotlivých instalačních disket lze získat i z veřejně přístupných elektronických archivů. V Praze se jedná o počítač `cs.felk.cvut.cz` zapojený v síti Internet. Pomocí programu ftp můžete získat přístup k archívu pod uživatelským jménem `anonymous` a heslem `anonymous`. Instalační diskety naleznete v adresáři `[pub.tex.cstex]` a jeho podadresářích. Výjimku tvoří disketa číslo 5, která je k dispozici pouze členům  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}\mathcal{U}$ . Instalaci  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u spolu s dalším relevantním software je také možné získat z elektronického archívu Masarykovy University v Brně na počítači `ftp.muni.cz`, uživatelské jméno `anonymous`. Jako heslo

udejte vaši emailovou adresu pro statistické účely. T<sub>E</sub>Xový software najdete v adresáři `pub/tex`. Chcete-li do archívu uložit další software, uložte jej programem ftp do adresáře `pub/incoming/tex` a pošlete o tom krátkou zprávu na adresu `tex-support@muni.cz`. Na tuto adresu se též obraťte v případě problémů s tímto archívem.

Ne každý potřebuje nutně pro své potřeby 18 disket. Minimální počet disket pro vytvoření provozuschopné instalace je 3. Např. diskety 1, 2, 6 pro uživatele, kteří pracují pouze s fonty 180 dpi k L<sup>A</sup>T<sub>E</sub>Xu a nebudou chtít instalovat tzv. rozšířené fonty. Jedná se o fonty, které bývají součástí instalací, ale pokud se explicitě nevolají příkazem `\font`, nikdy se při „normálním“ provozu nepoužijí. Nebo diskety 1, 2 a 4 pro ty uživatele, kteří si budou chtít sami generovat bitové mapy fontů pomocí METAFONTu. Podrobnější informaci o obsahu jednotlivých instalačních disket si lze přečíst v `install.doc` (první disketa).

Instalace probíhá selektivně spuštěním dávky `install.bat`. Instalační program nabízí velké množství nápověd, takže by mělo být umožněno instalovat T<sub>E</sub>X i nezasvěceným uživatelům. Instalace veškerého balíku (všech 18 disket) vyžaduje na počítači s procesorem 386/33 Mz zhruba hodinu strojového času: 20 minut se kopírují diskety a dále pracuje počítač sám bez obsluhy. Nicméně nepředpokládejte, že budete za hodinu hotovi. Instalace totiž položí při startu asi 60 otázek na to, co si přejete instalovat, takže počítejte s celkovou dobou strávenou nad instalací minimálně 2 hodiny a vybavte se trpělivostí. Instalujte nejprve raději méně než více, protože kdykoli můžete snadno provést doplňující instalaci chybějících modulů.

Instalační program je vybaven jistou odolností proti přeplnění disku. Pokud instalace havaruje pro nedostatek místa na disku, budete mít možnost uvolnit disk (např. vymažete nějaké své oblíbené hry). Pak lze spustit instalaci z místa, kde byla přerušena. Pro úspěšnou instalaci je potřeba zhruba 6 až 40 MB diskové paměti (podle požadavků, co chcete instalovat). Minimální funkční instalace se nakonec vejde do 5 MB: instalujete pouze plain nebo L<sup>A</sup>T<sub>E</sub>X jenom se základními fonty.

Při rozhodování o instalaci fontů (které zabírají nejvíce místa) vám pomůže v nápovědě k instalačním otázkám přehledná tabulka požadovaných diskových kapacit. Zde je třeba upozornit na to, že při sestavování knihoven (to se děje automaticky v rámci instalačního programu) je nutné mít na chvíli dvakrát tolik místa na disku, než je velikost výsledné knihovny. Například knihovna 300 dpi zabere při požadavku na instalaci *všech* fontů asi 8 MB. Při sestavování potřebujete tedy mít dalších 8 MB místa na disku. Při skromnějších podmínkách proto doporučujeme provést instalaci dvakrát: poprvé instalujte pouze fonty (a nic jiného) a při druhém spuštění instalace požadujte instalovat ostatní software.

Stručný text k ukončení instalace najdete v souboru `start\codelat.txt` a vlastní dokumentace k C<sup>S</sup>T<sub>E</sub>Xu je v souborech `doc\user.doc` a `doc\cfg.doc`. První soubor obsahuje uživatelskou dokumentaci a druhý technickou. Dokumentace k jednotlivým programům a ovladačům je v adresáři `doc\programs`. Tato dokumentace popisuje relativně podrobně přípravu T<sub>E</sub>Xu k vlastní práci.

Jelikož se hlouběji zabýváme pouze prací se souborem `plain.tex` (jde o známé základní makro, vytvořené D. Knuthem), připomeňme, že je nutné jej inicializovat. Tato věc se provede velmi snadno spuštěním inicializační dávky `start\initex.bat`. Dávka nejprve zobrazí nabídku, v níž lze vybrat, které soubory (formáty) chceme inicializovat, jaké vzory dělení slov se mají při inicializaci zahrnout, jaké budete používat kódování vstupních textů a zda bude použit „Big-TeX“ (v tomto balíku se jedná o `tex386`) či nikoli. Nakonec se automaticky provede vlastní inicializace.

V  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u najdete dále nabídku tří editorů. Instalační program nabízí editor TE (volně šířený, ale nepříliš dobrý), Qedit (shareware, tj. program, jehož demo-verze je zcela funkční, ale při rozhodnutí o trvalém užívání je nutné zaplatit licenci) a editor  $\mathcal{C}\mathcal{S}\mathcal{E}\mathcal{D}$ , který zakoupil  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}$  a je určen pouze pro členy sdružení. Lze též nakonfigurovat jakýkoli jiný editor (například Norton editor, nebo editor, který je součástí vyšších verzí DOSu).

V této souvislosti je vhodné si rozmyslet, jak budeme pracovat s akcentovanými znaky při pořizování vstupního textu v editoru. To samozřejmě závisí na volbě samotného editoru. Například posledně jmenovaný  $\mathcal{C}\mathcal{S}\mathcal{E}\mathcal{D}$  řeší práci s akcentovanými znaky ve své vlastní režii a nemusíte se o nic starat. V jiném případě je potřeba zavést v pravý okamžik do počítače dva programy: ovladač klávesnice a ovladač zobrazení na obrazovce. K tomu je možné využít software, který je zahrnut do instalace a pracuje v kódu Kamenických. Pokud jde o klávesnici, jedná se o program `kbd.com` (jehož zdrojový text v Assembleru byl publikován v Bajtu), a pokud jde o zobrazení na obrazovce, jedná se o soubor pro zavedení tzv. „stránky DOSu“. Jak se to dělá, si můžete přečíst v souboru `start\csfonts.txt`. Samozřejmě můžete do instalace snadno začlenit svůj vlastní oblíbený editor a vše může vypadat jinak.

Před prvním spuštěním  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u v podstatě není třeba provést zásahy ani do souboru `autoexec.bat`, ani do `config.sys`. Pouze je vhodné přezkontrolovat, zda je v `config.sys` nastavena dostatečně velká hodnota pro parametry `FILES` a `BUFFERS`. Doporučuje se pro oba parametry volit hodnotu aspoň 10. Dále se předpokládá, že ve slušných počítačích se nerozšiřuje cesta `PATH` v `autoexecu` do závratných velikostí, ale je tam obsažen adresář (například `c:\bat`), kde jsou uloženy všechny spouštěcí dávky pro různé systémy. Do tohoto adresáře si zkopírujte spouštěcí dávky  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u, které najdete v adresáři `\emtex\start`, a můžete přistoupit k tolik očekávanému spuštění  $\mathcal{T}\mathcal{E}\mathcal{X}$ u. Přejděte do pracovního adresáře (kde budou vznikat všechny soubory související s vaším dokumentem) a v DOSu napište

```
plain mujtext
```

kde `mujtext` je název dokumentu. Pro vstupní soubor se automaticky připojí přípona `tex` a *nesmí* být v příkazovém řádku napsána. Pokud už dokument existuje, příslušný vstupní soubor musí mít tuto příponu a žádnou jinou. V dřívějších verzích  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u se rozlišovalo mezi českým a anglickým dokumentem a existovala ještě přípona `cs`, ovšem v námi popisované verzi už tomu tak není. Původní přípona `cs` nebyla totiž podporována  $\mathcal{L}\mathcal{A}\mathcal{T}\mathcal{E}\mathcal{X}$ ovým příkazem `\include`, takže při členění do-

kumentu do více souborů docházelo k míchání dvou typů přípon a tím vznikaly zmatky při zpracování těchto souborů pomocnými rutinami.

Například v adresáři `start` najdete soubor `adjust.tex`, což je pokusný text pro seřízení okrajů tiskárny. Přejděte tedy do zmíněného adresáře a napište `plain adjust .`

Poznamenejme, že příkaz `plain` startuje systém  $\TeX$ u do režimu zpracování formátem `plain`, což je předmětem této knihy. Místo tohoto příkazu lze použít také příkazy `latex`, `amstex`, `aml`, `lams` a `slitex`.

Po vyzkoušení všech funkcí se doporučuje disk a adresář obsahující  $\TeX$ ovský software „srovnat“ nějakým programem typu Norton Speed Disk (vyšší verze DOSu mohou takovou práci provést automaticky). Také je vhodné zabezpečit disk proti zápisu neprivilegovaným uživatelem (nutné v síťovém provozu, případně při více-uživatelském použití). Prostředky pro toto zabezpečení nejsou v instalaci obsaženy a bohužel žádoucí zabezpečení neposkytuje ani DOS.

## 1.5 Pracujeme s $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ em

Uvedme nejprve jakousi doporučenou strategii v uschovávání rozpracovaných souborů. V předchozím odstavci bylo zdůrazněno, že veškeré soubory (`tex`, `dvi`, `log`, `aux` apod.) vznikají v *aktuálním* adresáři; to je adresář, z něhož vyvoláte systém  $\TeX$ . Co má uživatel ve svém adresáři, je pouze jeho věc a záleží na jeho „čistotnosti“, zda si po skončení práce po sobě uklidí, tj. vymaže nepotřebné pracovní soubory. Samotný systém totiž nemůže vědět, které soubory považuje uživatel za nepotřebné, a proto úklid souborů za něj neprovádí. Je pravda, že někteří administrátoři systému  $\TeX$  jej konfigurují tak, že nechávají uživateli v pracovním adresáři jen vstupní soubory a vše ostatní mažou. Tuto přílišnou péči nepovažujeme za vhodnou, a proto ve výchozí konfiguraci  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u není zahrnuta.

Mimo tento aktuální adresář existuje adresář se společnými textovými vstupy (většinou stylovými soubory), který je zahrnut do struktury adresářů systému. Bylo by ideální, kdyby tento společný balík stylových souborů *nebyl* svévolně rozšiřován o další „nálezy“ ve veřejných archívech. Nechtě taková rozšíření jsou uživatelům nabídnuta ke zkopírování do jejich vlastních adresářů, případně nechtě si uživatelé ve svých adresářích tyto soubory sami vytvářejí. Pak určitě budou přesně vědět, co dělají a nestane se, že dodají svým přátelům neúplný zdrojový text svého díla. To se zatím často stávalo. Příjemce zdrojového textu pak byl zaměstnán pátráním po nedodaném stylu, případně odhadováním, co asi ten styl mohl obsahovat a co je tedy třeba v jazyce  $\TeX$ u „naprogramovat“, aby dodaný zdrojový text byl vůbec zpracovatelný. Až se uživatel bude muset sám postarat o umístění všech nestandardních textových vstupů, tyto problémy odpadnou. Tyto nestandardní vstupy nemusí samozřejmě být jen v aktuálním adresáři, ovšem pak se uživatel musí postarat o příslušné rozšíření hledacích cest. Balík stylových souborů obsažených

v  $\LaTeX$ u pokládáme za standard a vše mimo něj by mělo být součástí vlastního dokumentu.

Z uvedené poznámky plyne, že pokud pracujete na větším dokumentu (například na své knize), pak je vhodné si pro něj vytvořit speciální adresář a v něm mít všechny textové vstupy včetně nestandardních stylů. Také je užitečné rozčlenit jednotlivé kapitoly do různých souborů, všechny ovšem ve společném adresáři. Takto vytvořené dílo pak snadno ve svém adresáři zkomprimujete pro účely archivace a nic v archívu nebude chybět. Do jednotlivých adresářů si můžete ukládat třeba i více menších dokumentů (například můžete mít adresář s osobní korespondencí apod.). Někteří preferují členění do adresářů podle použitého formátu (plain  $\TeX$ ,  $\LaTeX$  apod.), jiní (např. písárky) mají soubory členěné podle zakázek.

Nyní popíšeme vlastní práci s  $\LaTeX$ em. Nejprve tedy přejdete do pracovního adresáře, který je prázdný nebo obsahuje rozpracovaný dokument. Pak odstartujete systém například příkazem `plain mujtext`, kde `mujtext` je vámi zvolený název dokumentu (příponu `tex` nepište). Na obrazovce se objeví nabídka (menu). Nej důležitějšími položkami jsou položky s názvy `Edit`, `PlainTeX` a `View`. Klávesa `F1` poskytuje nápovědu ke zrovna zvýrazněné položce.

Položka `Edit` spustí editor, který jste si vybrali při instalaci. Tuto volbu editoru je možno (podobně jako spoustu dalších věcí) ještě dodatečně měnit z „uživatelské pozice“ (tedy bez zásahu do části disku obsahující  $\TeX$  a případně zabezpečené proti zápisu). Například při víceuživatelském provozu může každý uživatel mít nakonfigurován svůj *vlastní oblíbený* editor. Jak se to dělá, si může uživatel přečíst v souboru `doc\user.doc`.

Položka `PlainTeX` spustí vlastní formátování dokumentu (říkáme též „překlad“ vstupního souboru  $\TeX$ em). Výstupem této činnosti bývá většinou chybové hlášení. Jste-li úspěšnější, pak si můžete prohlédnout výsledný dokument prohlížečem, který spustíte položkou `View`. V instalaci použitý prohlížeč ovládáme šipkami, klávesami `PgDn`, resp. `PgUp` (pro skok na další, resp. předchozí stránku) a klávesa `Q` prohlížení končí. Další možnosti prohlížeče si přečtete v nápovědě v hlavní nabídce („najeďte“ na položku `View` a zmáčkněte klávesu `F1`) nebo můžete nahlédnout do velmi rozsáhlé dokumentace v souboru `doc\programs\dvidrv.doc`.

V případě výskytu chyb je nutné se vrátit zpět do editoru. Na  $\TeX$ ovský ne příliš sympatický prompt tvaru „?“ je možno odpovědět písmenem `e` pro přímý vstup do editoru nebo písmenem `x` pro ukončení běhu  $\TeX$ u a vstup do editoru prostřednictvím položky `Edit` v nabídce. První alternativa se zdá lepší, ovšem je třeba si uvědomit, že  $\TeX$  i zde ukončí svou činnost a po opravě chyby začne zpracování znovu od začátku. Je proto velmi vhodné odpovědět na zmíněný otazník jednoduše klávesou `Enter` pro přeskočení chyby, přičemž  $\TeX$  může najít další. Také se používá písmeno `s` nebo `q` pro přeskočení i všech následujících chyb. Je-li  $\TeX$ ovský prompt ve tvaru `*`, znamená to, že  $\TeX$  ukončil čtení vstupního souboru a nyní chce číst další příkazy rovnou z terminálu. To se stává v případě, že vstupní soubor neobsahuje ukončovací příkaz `\bye` nebo `\end`. V takovém přípa-

dě je vhodné odpovědět příkazem `\end` a příkaz do vstupního souboru pro příště doplnit. Ještě se může objevit jedna nepříjemnost: `TeX` se ptá na jméno souboru. Vstupní soubor (případně jiný, začleněný příkazem `\input`) nebyl nalezen. Zde pro ukončení nepomůže ani `Enter`, ani `\end` (`TeX` se ptá tvrdšíjné znovu), ale zabírá kombinace `Ctrl-Z` nebo můžeme nabídnout soubor `nul`.

Při zpracování vstupního souboru `TeXem` se kromě `dvi` souboru vytváří soubor s příponou `log`, v němž je protokol o zpracování včetně případných chybových hlášení. Po chybném překladu se pak při vyvolání editoru objeví kurzor v *řádku* (tedy nikoli přesně v *pozici*) výskytu první chyby, přičemž editor pracuje ještě s jedním oknem, kde je zobrazen soubor s chybami (soubor `log`) a kde vidíte chybové hlášení. Příslušnou klávesou v editoru (to záleží na editoru) lze „skákat“ na další výskyty chyb. Upozorňujeme, že zmíněná vlastnost závisí na konfiguraci editoru v systému a měla by fungovat pro všechny tři editory, které instalační program `CSTeXu` nabízí. Instalujete-li svůj vlastní editor, musíte se o to sami postarat čtením dokumentace k editoru a k `CSTeXu` (soubor `doc\user.doc`). Může se stát, že tuto vlastnost vám váš editor neumožní.

I v případě, že je vstupní soubor bez chyb, objeví se v souboru s příponou `log` a případně i na obrazovce řada hlášení. Takové hlášení může mít např. tvar:

```
This is emTeX, Version 3.14 [3c-beta5]
      (preloaded format=csPLAIN 93.5.29) 1 OCT 1993 23:17
**&csPLAIN intro.tex
(intro.tex
\hsize=159.2 mm \vsize=239.2 mm
(c:\emtex\texinput\czech.sty
Document Style Option 'czech' Version 1.141, April 26, 1993.
Czech hyphenation used. \frenchspacing is set on.)
\contents=\write0
\index=\write1
\exno=\count33
....
\minute=\count43
[1] [-1] [-2] [-3] [-4] [-5] [-6] [1] [2] [3] ..... [98] [99]
[100] [101] [102] [103] )
Output written on intro.dvi (110 pages, 338164 bytes).
```

Můžeme z něj lehce vyčíst, že šlo o zpracování `emTeXem` verzí 3.14, formát `csPLAIN` byl inicializován 29. května 1993 a dokument byl zpracován 1. října 1993 v 23 hodin a 17 minut. Vstupní soubor se jmenoval `intro.tex` (šlo o zpracování jedné verze této knížky) a byl uložen v aktuálním adresáři. Byl zpracován pomocí makra `czech.sty` pro úpravu češtiny. Bylo použito „francouzského stylu“, tj. za tečkami na konci vět apod. se vkládala normální, nezvětšená mezera. Byly použity vzory dělení slov pro češtinu. Vzniklo celkem 110 stránek textu a soubor `intro.dvi` byl uložen do aktuálního adresáře.

Nyní se porozhlédneme po dalších položkách v hlavní nabídce (menu)  $\text{\TeX}$ . Nemá smysl podrobně popisovat všechny možnosti nabídky, protože jejich význam lze kdykoli zjistit z nápovědy (F1) a navíc je to věc, která značně podléhá nápadům „ $\text{\TeX}$ ovského inženýra“, který konfiguraci pro vás (například na pracovišti) připravil. Nabídka založená na programu MNU je totiž plně konfigurovatelná. Znamená to, že vše, co vidíte na obrazovce v prostředí nabídky, závisí na lokální konfiguraci  $\text{\TeX}$ . Dokumentaci k tomu jak konfigurovat nabídku najdete v souboru `doc\programs\mnu.tex`.

Zůstaňme nejprve v hlavním okně  $\text{\TeX}$ . Zde najdete kromě tří zmíněných a nej-používanějších položek také další. Například položkou `Spell` vyvoláte korektor překlepů a položka `Tie` (vlnka) spustí program `vlnka.exe`, který do zdrojového souboru doplní za všechna jednopísmenná slova `k,K,o,O,s,S,u,U,v,V,z,Z` vlnku.

Položka `Others` otevře další okno s nabídkami na spuštění některých doplňkových programů a položka `CStoCS` nabídne možnost spustit konvertor `cstocs.exe`. Tento program vám umožní transformovat vstupní texty s diakritikou mezi šesti nejběžnějšími kódováním (jsou zahrnuty jen znaky z české a slovenské abecedy) a navíc je schopen tyto texty převádět do  $\text{\TeX}$ ovské transkripce a zpět. Bude se vám hodit například tehdy, když dostanete text v jiném kódování, než je to, které máte instalováno v systému.

V levém okně `File` lze vyvoláním položky `Name` změnit název dokumentu a pracovat na něčem jiném (ovšem stále ve stejném adresáři). Nemělo by vás zaskočit, že zde můžete zadat dva různé názvy souboru. Nápověda vám vysvětlí, že to je vhodné při práci na rozsáhlém díle, kdy máte například jednotlivé kapitoly v různých pracovních souborech a v hlavním souboru jsou pouze volací příkazy pro tyto kapitoly.

Také můžete vyvolat řádku DOSu, aniž byste opustili prostředí hlavní nabídky, případně lze z nabídky vyvolat program typu Norton Commander, Xtree apod. pro správu souborů. Zde záleží na uživateli, který program nejraději používá a který si nakonfiguruje. Také lze vyvolat jednoduchou kalkulačku a položka `Quit` definitivně ukončí práci s nabídkou  $\text{\TeX}$ ovských programů.

V okně `Print` najdete položku na nastavování parametrů ovladačů pro tisk (`Options`) a položky na poslední prohlédnutí výsledku před tiskem (označené slovem `Preview`). Prohlížeč spuštěný z těchto položek zahrne i vámi zadané tiskové parametry a jsou použity stejné fonty, jako půjdou do tisku). Položky `Print Laser`, `Print Matrix` a `Print 9 Matrix` spouštějí vlastní tisk na příslušné tiskárně. Také zde najdete položky, které spustí „tisk do souboru“, tj. vznikne soubor, který pak dodatečně pouhým binárním přeprogramováním do tiskárny způsobí vlastní tisk. Této možnosti využijete, pokud chcete přenést dokument k počítači, který není vybaven žádnou instalací  $\text{\TeX}$ , ale je vybaven tiskárnou.<sup>1</sup> Nevýhodou je, že pokud při práci na tomto počítači zjistíte jakékoli problémy (např. se mechanicky poškodí některá

---

<sup>1</sup> Máte-li „objemný dokument“, může se stát, že se vám tiskový soubor nevejde na disketu. Formát těchto souborů totiž nešetří příliš místem, ale právě proto je



stránka), musíte tisknout celý dokument znovu. Tato nevýhoda při použití tisku rovnou z instalace  $\text{T}_{\text{E}}\text{X}$ u odpadá, protože můžete nastavit parametry tiskových ovladačů například tak, že tisknete jen některé stránky.

V okně `Options` lze změnit režim zpracování podle formátu (plain  $\text{T}_{\text{E}}\text{X}$ ,  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ , atd.), druh použitého programu `tex.exe` (zda `Big- $\text{T}_{\text{E}}\text{X}$`  či `nikoli`) a položka `Envir` je určena k nastavení uživatelem definovaného chování konfigurace (volba editoru, parametry pro  $\text{T}_{\text{E}}\text{X}$  a prohlížeč a mnoho dalších věcí).

Nakonec v okně `METAFONT` najdete nástroje na pořizování obrázků a jednotlivých fontů programem `METAFONT`. Tento skvělý program vytvořil rovněž D. Knuth a slouží k návrhu nových písem a digitalizaci fontů do stanovených rozlišení. Na tomto místě uvedeme jen velmi krátkou informaci o programu `METAFONT`. Čtenář se asi v tuto chvíli bude chtít naučit především základy  $\text{T}_{\text{E}}\text{X}$ u (proto si pořídil tuto knížku) a je možné, že odbočka k `METAFONT`u by ho jen zdržovala. Pokud tedy zrovna nechcete vytvářet nové fonty a navíc se spokojíte s těmi fonty, které máte už v instalaci digitalizované, nebudete `METAFONT` potřebovat. V takovém případě je možné přeskocit zbytek tohoto odstavce. Můžete se sem vrátit později, až budete mít první zkušenosti s  $\text{T}_{\text{E}}\text{X}$ em za sebou.

Vstupem pro `METAFONT` je textový soubor, který vektorově popisuje tahy jednotlivých písem. Protože tento soubor vlastně řídí činnost `METAFONT`u, říkáme mu též zdrojový program daného fontu a `METAFONT` pak přebírá roli jakéhosi „překladače“ podobně, jako je tomu při zpracování dokumentu  $\text{T}_{\text{E}}\text{X}$ em.

Balík  $\mathcal{C}\mathcal{S}\text{T}_{\text{E}}\text{X}$ u obsahuje nejen program `METAFONT` a další potřebné programy, ale i zdrojové programy všech `Computer Modern` řezů, které vytvořil D. Knuth. Kromě nich zde najdete též zdrojové texty  $\mathcal{C}\mathcal{S}$ -fontů, které se v dolní polovině kódovací tabulky shodují s `Computer Modern` a ve zbytku tabulky obsahují kompletní akcentovaná písmena pro českou a slovenskou sazbu. Jsou zde i méně běžné fonty, které byly použity třeba jen při sazbě knih **The  $\text{T}_{\text{E}}\text{X}$ book** a **The METAFONTbook**.

Dále jsou v  $\mathcal{C}\mathcal{S}\text{T}_{\text{E}}\text{X}$ u k dispozici zdrojové programy všech symbolů a písem, jež jsou součástí  $\mathcal{A}\mathcal{M}\mathcal{S}\text{-T}_{\text{E}}\text{X}$ u; kromě rozšířených možností pro sazbu různých matematických symbolů jsou v něm i krásná písmena vytvořená významným typografem Hermanem Zapfem (ovšem pozor – nejedná se o textová písmena, ale o písmena použitelná v matematické sazbě, tj. programy neobsahují kerning ani diakritická znaménka). Z veřejných archívů můžete získat mnoho dalších písem v podobě zdrojových souborů pro `METAFONT`. Najdete tam například všechny běžné řezy azbuky, různá písmena pro speciální použití apod.

Ke generování fontů (nejspíše ze zdrojových souborů obsažených v  $\mathcal{C}\mathcal{S}\text{T}_{\text{E}}\text{X}$ u) potřebujete především program `METAFONT` (`mf.exe`), dále program `gftopk.exe`, základní soubor `maker plain.mf` a ke `Computer Modern` také další makra v sou-

---

použití komprimačních programů (`zip`, `arj`, atd.) vysoce účinné a většinou problém vyřeší.

boru `cm.mf`. Důležitý je také soubor `local.mf`, v němž jsou definovány parametry běžných výstupních zařízení (zejména hustota v bodech na palec – dpi).

Všechny zmíněné softwarové prostředky se při instalaci  $\LaTeX$ u (po potvrzení, že chcete instalovat METAFONT) samy nainstalují do správných adresářů a navíc se zajistí požadované naplnění všech proměnných obsahující „vyhledávací adresáře“ pro příslušné programy a soubory. V závěru instalace se také automaticky vygenerují báze `plain.bas` a `cm.bas`, takže uživatel nemusí zpočátku o METAFONTu nic vědět.

V okně METAFONT/Parameters zadáte nejprve název souboru (přípona `mf` se doplní sama), se kterým chcete pracovat, tj. editovat a případně na něj spouštět METAFONT. Zkuste napsat třeba `logo10`. Jedná se o font, obsahující písmena M, E, T, A, F, O a N, tedy písmena pro logo METAFONT v desetibodové velikosti.

V dalším řádku jsou parametry pro spouštění METAFONTu. Nejčastěji tam bývá napsáno

```
&plain \mode=hplaser; mag=1.0; input %MF%,
```

což znamená, že METAFONT bude pracovat sází `plain` (podobně, jako v  $\TeX$ u existují formáty). Parametry výstupního zařízení (rozlišovací schopnost atd.) jsou zde definovány ve funkci `hplaser` (najdete ji v souboru `local.mf`), faktor zvětšení fontu je roven jedné (`mag=1.0`) a vstupním souborem bude soubor definovaný jako pracovní (v tomto případě soubor `logo10.mf`). Za parametrem `mag` zkuste například změnit zvětšení – lze zde použít libovolné desetinné číslo a vygenerovat například písmo „náštěnkové velikosti“.

Nyní přijde nejkomplicovanější část. METAFONT ukládá úspěšně vygenerovaný font do dvou souborů. V našem případě vznikne soubor `logo10.tfm` (tam jsou metrické údaje pro  $\TeX$ ) a `logo10.#gf`, kde symbolem `#` je označeno číslo udávající efektivní rozlišení fontu. Jedná se o takové rozlišení, které vznikne zmenšením fontu (například při dodatečném zmenšení předlohy na kopírce) do původní velikosti fontu jedna ku jedné. Například, pokud jsme použili `mag=1.2` a `mode=hplaser` (tedy 300 dpi), pak efektivní rozlišení bude  $1.2 \cdot 300 = 360$  a na výstupu tedy dostaneme soubor `logo10.360gf` (v MS DOSu ovšem písmena `gf` v extenzi neuvídíme). Tento soubor obsahuje digitální mapy písmen a je nutné jej konvertovat do tvaru čitelného tiskovými ovladači, tj. do souboru s příponou `pk`.

Ke konverzi do `pk` formátu se použije program `gftopk.exe`. V našem příkladě bude potřeba použít následující parametry.

```
gftopk logo10.360 logo10.pk
```

Soubor `logo10.pk` vznikne v aktuálním adresáři, kde jej všechny ovladače najdou. Můžeme tedy nový font hned použít v  $\TeX$ u zaváděcím příkazem `\font` uvedeným většinou na začátku zdrojového souboru:

```
\font\mflogo=logo10 scaled 1200
```

nebo

```
\font\mflogo=logo10 scaled \magstep1 ,
```

což je totéž. Poznamenejme, že tisíckrát parametr `mag` v METAFONTu rovná se celočíselný parametr za slovem `scaled` v T<sub>E</sub>Xu (ovšem jen v případě, že nemáte nastaveno v T<sub>E</sub>Xu globální zvětšení celého dokumentu příkazem `\mag` nebo `\magnification`). Dále slovo `\magstep`  $k$  je v T<sub>E</sub>Xu definováno jako číslo  $1, 2^k \cdot 1\,000$  pro  $k = 0, 1, 2, \dots, 5$ .

Pokud máte přístup do instalace C<sub>S</sub>T<sub>E</sub>Xu (disk není pro vás zabezpečen proti zápisu), pak je vhodné vyrobený soubor `pk` uložit do příslušného adresáře podle efektivního rozlišení fontu. Kdybyste totiž vyrobili další font stejného jména, ale jiné velikosti v aktuálním adresáři, pak by se původní font přemazal. Ovladače přitom umějí hledat fonty (například pro laserovou tiskárnu s 300 dpi) v souborech

```
\emtex\fonts\pk300\dpir\*.pk ,
```

kde  $r$  je efektivní rozlišení fontu a `\emtex` je hlavní adresář systému s T<sub>E</sub>Xem (může se jmenovat jinak). V našem příkladě tedy můžeme výstup z `gftopk` směřovat rovnou do příslušného adresáře příkazem

```
gftopk logo10.360 δ:\emtex\fonts\dpir\logo10.pk ,
```

kde  $\delta$  je disk s C<sub>S</sub>T<sub>E</sub>Xem. Na tomto místě je vhodné upozornit, že bitové mapy v instalaci C<sub>S</sub>T<sub>E</sub>Xu jsou v jiném formátu, tzv. formátu knihoven fontů (soubory s příponou `fli`). Do instalace je zahrnut program `fontlib.exe`, který tyto knihovny ze souborů `pk` vytváří. Tento program ovšem nemusíte použít, protože ovladače umí číst jak ze souborů `fli`, tak ze souborů `pk`, a to současně. Asi je dobrá strategie nechat v knihovnách `fli` pouze bitové mapy, které jsou součástí dodávky C<sub>S</sub>T<sub>E</sub>Xu, a nové bitové mapy pro přehlednost ponechat ve tvaru `pk`.

A jak to je se souborem `tfm`? Ten bude pro všechna zvětšení fontů stejný a bude obsahovat rozměrové údaje pro font, jako by byl použit parametr `mag=1`. Teprve při jeho zavedení si tyto údaje T<sub>E</sub>X sám pronásobí koeficientem definovaným za slovem `scaled`. Proto jej můžeme nechat v aktuálním adresáři, případně jej zahrneme do systému do adresáře `\emtex\tfm`. Ovšem většinou už tam takový soubor najdeme a jeho přemazávání novou verzí je zbytečné.

Jiná situace je při tvorbě obrázků METAFONTem (co „písmeno“, to vlastní vámi naprogramovaný obrázek). Pak bývá vhodné jak soubor `tfm`, tak `pk` ponechat v aktuálním adresáři, kde bývají rozpracovány i další soubory, týkající se pouze vašeho dokumentu. Zde snaha o zařazení těchto souborů do adresářové struktury s C<sub>S</sub>T<sub>E</sub>Xem je poněkud postavená na hlavu, protože žádný jiný uživatel váš font (obrázky) nepoužije.

Velmi užitečný pro generování nových řezů písem je Mattesův program `mfjob`, který je rovněž obsažen v balíku C<sub>S</sub>T<sub>E</sub>Xu. Ten umožňuje nepřerušované generování většího množství různých písem tak, že sám zařídí vše potřebné: běh METAFONTu,

přeměnu vzniklých generických souborů na `pk` formát a odeslání souborů `pk` a `tfm` do určených adresářů. Nemusíme vůbec nic počítat, protože `mf job` spočítá efektivní rozlišení fontu za nás. Popisem fungování `mf job` nebudeme čtenáře zdržovat od studia `TeX`, protože součástí `CTeX` je i podrobná dokumentace.

V neposlední řadě upozorníme na velmi důležitou vlastnost Mattesových ovladačů (pro tisk i prohlížení). Jedná se o možnost nakonfigurovat je tak, aby při nenalezených fontech v instalaci v podobě bitových map slušně tuto skutečnost oznámily a zeptaly se, zda situaci náhodou nechcete napravit automatickým vyvoláním `mf job`. Odpovíte-li `y` (tedy ano), pak se už jen díváte na obrazovku, jak se opakovaně volá `METAFONT` a `gftopk` na všechny chybějící fonty a nakonec se znovu spustí vyvolaný ovladač, ovšem s nenalezenými fonty už nemá problémy. Tato věc není implicitně v `CTeX` konfigurována, protože zde záleží hodně na tom, zda ponecháte disk s `CTeX`em z těchto důvodů otevřený k zápisu libovolným uživatelem (systém vám bude dynamicky růst a zabírat stále více paměti), nebo se rozhodnete pro nějaký „přechodný“ adresář, kam se tyto fonty budou vytvářet (např. mimo disk s `CTeX`em) a periodicky budete zvažovat (z pozice správce systému), zda fonty z tohoto přechodného adresáře překopírujete do systému či smažete. Informaci o zmíněném konfigurování ovladačů najdete v souboru `doc\cfg.doc`

`TeX` ve spojení s `METAFONT`em je opravdu mocný prostředek, který umožňuje nejrůznější kouzla včetně rotací písmen a dalších efektů. Zvládnutí takových věcí už ovšem vyžaduje přinejmenším nějaký „jemný úvod do `METAFONT`u“, který bohužel zatím neexistuje.

## 1.6 Čeština a slovenština v `CTeX`u

Pozastavme se nejdříve nad mechanismem práce s akcenty. Bohužel z důvodu požadavku na existenci kapitol o `CTeX`u zcela zvlášť bez zásahu do následujícího originálního Doobova textu se zde budeme muset odvolávat na věci, které se čtenář naučí až za chvíli v následujících kapitolách. Pro první čtení vám přitom stačí ujistění, že v editoru píšete normálně s akcenty (tak, jak vám to váš editor umožní) a nikoli pomocí odstrašující transkripce akcentů, jak se s ní seznámíte v dalších kapitolách. S tímto vědomím můžete následující řádky o akcentech v `CTeX`u přeskóčit a vrátit se k nim později.

Při inicializaci formátu jste například nakonfigurovali kódování vstupních textů podle Kamenických. Napíšete-li nyní do vstupního textu třeba písmeno `ř`, pak v editoru je toto písmeno zpracováno a zobrazeno podle kódování Kamenických. Program `TeX` toto písmeno přečte a okamžitě transformuje do vnitřního kódu `TeX`u, který uživatele nemusí zajímat. Ve výstupním souboru `dvi` je odkaz na písmeno `ř` podle tohoto vnitřního kódu, v němž jsou též kódovány `CS`-fonty. Ovladač pak při tisku nebo při prohlížení „vyloví“ z `CS`-fontů písmeno `ř` z patřičného místa.

Takto bezvadně to ale bude fungovat pouze při použití tzv. CS-formátů a při požadavku sazby písmene ř fontem definovaným ve formátu. Protože v  $\mathcal{CS}\TeX$ u se pracuje pouze s CS-formáty (tedy `csplain`, `cslatex` apod.), nemusíte se o nic starat. V kapitole 2.4 se naučíte používat příkazy `\it`, `\tt`, `\bf` a další. Tyto příkazy vám budou bez problémů s akcentovanými písmeny fungovat. Na druhé straně ale pozor na příkaz `\font`! Kapitola 2.4 vás naučí zavést vlastní font například příkazem

```
\font\bigrm = cmr10 scaled \magstep 1 .
```

Pak vám ale *nebudou* fungovat ve fontu `\bigrm` písmena s akcenty a v chybovém souboru (`log`) dostanete například zprávu ve tvaru:

```
Missing character ř in font cmr10 .
```

Původní Computer Modern fonty (mají v názvu na začátku písmena `cm`) jsou totiž určeny pouze pro anglický text a akcentovaná písmena lze skládat v těchto fontech pomocí příkazů v  $\TeX$ u tak, jak to je uvedeno v kapitole 2.2. Při použití akcentovaných znaků přímo musíme místo fontu `cm*` zavést font `cs*`, takže v našem příkladě použijeme

```
\font\bigrm = csr10 scaled \magstep 1 .
```

Druhá (méně doporučená) možnost je použít u „ručně“ zavedených fontů pro akcenty transkripci pomocí  $\TeX$ ovských sekvencí (viz kapitola 2.2). Tento způsob je ale jediným řešením pro psaní článku (například v angličtině), který potřebujeme poslat do zahraničí. Tam samozřejmě nejsou na žádné kódování podle Kamenických ani na  $\mathcal{CS}$ -fonty zvědaví. Chcete-li se v takovém článku podepsat a máte ve svém jménu písmeno s háčkem nebo čárkou, pak musíte použít mezinárodně uznávanou transkripci akcentů podle kapitoly 2.2. Tuto transkripci pak použijete i v citacích, kde se to může hemžit akcenty z celého světa.

Transkripce naopak nelze úspěšně použít pro plný český text, i když v ní budete zběhlí (tak jako první průkopníci  $\TeX$ u u nás) nebo užijete konvertor `CStoCS`, který umožní konverzi z definovaného kódování do transkripce  $\TeX$ ovými sekvencemi a zpět. Důvod je prostý: při použití transkripce přestává fungovat algoritmus pro dělení slov, takže všechna slova s akcentem by  $\TeX$  chápal jako nedělitelná. To mu podstatně znesnadní formátování odstavců.

Můžete si také posílat české zdrojové texty elektronickou poštou (mail). Tato služba ovšem akceptuje pouze text **bez hacku a carek**. Proto musí odesílatel použít konvertor `CStoCS` z definovaného kódování do transkripce a takový soubor může poslat. Příjemce pak může buď soubor rovnou zpracovat  $\TeX$ em (nebude mu ale fungovat dělení slov), nebo (lépe) nejprve soubor konvertuje programem `CStoCS` zpět do užívaného kódování a pak teprve použije  $\TeX$ . Uvedený způsob přenášení souborů s diakritikou pomocí elektronické pošty není příliš běžný. Častěji se pro tyto účely používají veřejně dostupné programy `uuencode` a `uudecode`, které soubory

transformují do nečitelné podoby a zpět, přičemž tato nečitelná podoba obsahuje pouze standardní ASCII znaky (tj. s kódy menšími než 128).

Dále se budeme zabývat soubory s makry (definicemi) pro  $\text{T}_{\text{E}}\text{X}$ , které usnadňují českou a slovenskou sazbu. Jedná se o sazbu národních uvozovek, přepnutí algoritmu na dělení slov do češtiny nebo slovenštiny, jiný než americký způsob mezerování mezi slovy a za větou a automatické generování slov typu „Kapitola“, „Obrázek“ namísto slov „Chapter“, „Picture“ apod. v  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ u. Také příkaz `\today` vysází datum v národním jazyce.

Mezi soubory  $\mathcal{C}\mathcal{S}\text{T}_{\text{E}}\text{X}$ u jsou takové soubory dva: `czech.sty` a `slovak.sty`. Oba se liší jen v detailech a v tom, že např. při použití souboru `slovak.sty` místo slova „Úvod“ dostaneme slovo „Predhovor“ a zapne se slovenské dělení slov. Za zmínku stojí, že lze načíst oba stylové soubory naráz (bez závislosti na pořadí) a pak příkazy `\czechTeX`, `\slovakTeX` a `\originalTeX` budou přepínat mezi třemi jazyky a to i uvnitř jediného odstavce.

Zde je vidět, že od verze 3.0 se  $\text{T}_{\text{E}}\text{X}$  stává skutečně mezinárodním. Samozřejmě je nutné, aby všechny použité jazyky měly své dělení slov zahrnuto ve formátu při inicializaci. Pokud budete chtít použít další jazyky, musíte si sehnat vzory dělení a stylový soubor pro příslušný jazyk a do zpracování dokumentu je zahrnout. Můžete bohužel narazit na problémy s nekompatibilitou těchto stylových souborů navzájem a s nekompatibilitou vnitřního kódování  $\text{T}_{\text{E}}\text{X}$ u. Problém je v tom, že vnitřní kódování  $\text{T}_{\text{E}}\text{X}$ u navržené v Corku pro evropské jazyky zasahuje do původního Knuthova návrhu (při použití Computer Modern fontů). Nyní lze z veřejných archívů sice získat tzv. systém „Babel“, což je kolekce stylových souborů pro evropské jazyky, která se opírá o kódování podle Corku, *ale* toto kódování skoro nikdo nepoužívá v plném rozsahu (minimálně je upraveno tak, aby dolní polovina tabulky kódování byla shodná s Computer Modern fonty). V  $\mathcal{C}\mathcal{S}\text{T}_{\text{E}}\text{X}$ u je pro vnitřní kódování použito kódování podle Computer Modern fontů v dolní polovině tabulky a rozšíření je podle kódu ISO-Latin2, protože to je standard pro střední a východní Evropu používaný na výkonných počítačích.

Popíšeme ještě soubor `czech.sty` z dílny O. Ulrycha. Tento soubor obsahuje předně některá řídicí slova pro snadný přechod k českým, resp. anglickým vzorům pro dělení slov. Příslušná řídicí slova (uvádíme je níže) je možné používat kdykoliv za řádkem

```
\input czech.sty
```

(nebo je-li jméno souboru `czech` uvedeno jako styl v  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ u, tj. začíná-li dokument v  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ u řádkem

```
\documentstyle[czech]{anystyle}).
```

Zavedení tohoto souboru automaticky inicializuje české dělení slov a rovnoměrné mezerování mezi slovy (na rozdíl od amerického způsobu mezerování, při němž mezery za interpunkcí se liší). Řídicí slova zavedená v tomto souboru jsou následující:

<code>\clqq</code> a <code>\crqq</code>	vysází české uvozovací a ukončovací uvozovky.
<code>\clq</code> a <code>\crq</code>	vysází české uvozovací a ukončovací apostrofy.
<code>\selectlanguage{n}</code>	přepíná do jazyka specifikovaného pomocí <code>n</code> , kde <code>n</code> může být jedno z následujících řídicích slov: <code>\czech</code> , <code>\english</code> a <code>\USenglish</code> ; toto mění formát <code>data</code> , prostředí (v $\text{\LaTeX}$ u) a (jestliže je instalován multilinguální $\text{\TeX}$ – což v našem případě je) příslušná pravidla dělení slov.
<code>\originalTeX</code>	uvádí vše do původního tvaru.
<code>\czechTeX</code>	nastavuje české dělení a prostředí (toto je implicitní nastavení při zavedení stylu <code>czech.sty</code> ) a v $\text{\LaTeX}$ u způsobí výstup českých slov pro záhlaví.
<code>\csprimeson</code>	aktivuje dvojznaky ‘ ‘ a ’ ’ tak, že budou chápány jako české uvozovky (totéž platí pro apostrofy při ‘ a ’).
<code>\csprimesoff</code>	nastavuje ‘ ‘, ’ ’, ‘ a ’ na jejich původní význam (anglických apostrofů a uvozovek).
<code>\csprime</code>	(= <code>\csprimeson</code> ) kvůli kompatibilitě s dřívější verzí.
<code>\cstieon</code>	způsobí, že znak <code>~</code> nebude v matematickém režimu nahrazován mezerou.
<code>\cstieoff</code>	způsobí, že znak <code>~</code> se bude chovat tak, jak je v $\text{\TeX}$ u původně navržen, tj. vždy se nahradí mezerou.
<code>\uv</code>	je řídicí slovo pro sázení českých uvozovek. Text, který má být vysázen v uvozovkách následuje za tímto řídicím slovem ve složených závorkách.

Podrobnější poznámky:

Pokud jste zvyklí používat pro uvozovky dvojice levých a pravých apostrofů, a nechcete tento zvyk měnit ani pro český jazyk, můžete na začátek dokumentu napsat řídicí slovo `\csprimeson` a v tomto případě budou levé apostrofy transformovány na české uvozovací apostrofy či uvozovky (pokud budou dva za sebou) a pravé apostrofy budou transformovány na české uzavírací apostrofy či uvozovky (pokud budou dva za sebou). Pak ovšem pozor na skutečné apostrofy ve slovech *l' Hospital*, *d' Artagnan*, apod.

Pokud budete používat program `vlnka` k doplnění předložek za jednopísmenná slova, je potřeba mít na paměti, že program nechápe gramatický význam písmen, a tudíž nerozezná jednopísmenná slova ve významu předložek a matematických proměnných, např. písmeno "v" ve významu předložky od téhož písmene použitého jako matematická proměnná. Naštěstí však zpravidla matematické proměnné nebývají psány samostatně (s mezerou před i za písmenem). Problém, o který jde, spočívá v tom, že  $\text{\TeX}$  nahrazuje `vlnku` mezerou vždy, a to i v matematickém textu. Pokud bude na začátku souboru uvedeno řídicí slovo `\cstieon`, pak `vlnka` v matematickém textu (která se tam pravděpodobně dostala po použití programu `vlnka`) nahrazována mezerou nebude.

## 1.7 Jdeme na to!

Z našeho hlediska je to tedy tak, že potřebujeme vytvořit vstupní  $\TeX$ ový soubor, z kterého vznikne čitelný výstup. Jak ale takový vstupní soubor vypadá? Skládá se ze symbolů, které lze nalézt na klávesnici terminálu nebo dálkopisu, tj. z písmen velké a malé abecedy, číslic, obvyklých interpunkčních a diakritických znamének a dalších znaků (ty tvoří obyčejný ASCII<sup>1</sup> kód). Většinu textu píšeme prostě normálním způsobem. Zvláštní instrukce obvykle začínají jedním z několika speciálních symbolů jako např. `#` nebo `&` (později je popíšeme podrobněji). Takto vypadá příklad  $\TeX$ ového vstupního souboru (nazvěte ho např. `mujtext.tex`):

```
Toto je moje prvn\`i filologicky nen\`aro\v cn\`a
\v eta v \TeX u.\bye
```

Na tomto místě ještě připomínáme, že takto odstrašující tvar pro český text ve skutečnosti není potřeba psát (a není to ani vhodné – viz předchozí odstavec o češtině a slovenštině v  $\LaTeX$ u). Proto budeme ukázky českých vstupních textů uvádět v čitelnější podobě (tak se také píše při práci s popsanou verzí  $\LaTeX$ u); např. předchozí vstupní text budeme psát ve tvaru

```
Toto je moje první filologicky nenáročná věta v \TeX u. \bye
```

Všimněte si nejprve, že písmena v této ukázce vypadají jako z psacího stroje. Tohoto písma užíváme ve všech příkladech, které budou ilustrovat vstup z klávesnice počítače nebo terminálu. Dále si všimněte zpětného lomítka (backslash) v textu (i když se vyskytuje v přepsané ukázce jen dvakrát, budete se s ním často setkávat). Brzo poznáte, že to je jeden z těch speciálních symbolů, o nichž jsme již mluvili, a že se v  $\TeX$ ových souborech používá velice často. Konečně pomocí `\bye` jsme ukončili text, aniž by se něco takového objevilo v tištěné podobě. Vytvořte nyní soubor, který obsahuje text příkladu, a použijte  $\TeX$  k vytvoření souboru `*.dvi` a ovladač, abyste si mohli prohlédnout, co jste vytvořili. Mělo by to vypadat takto:

```
Toto je moje první filologicky nenáročná věta v  $\TeX$ u.
```

Na konci stránky se objeví číslo stránky. Pokud jste se dostali tak daleko, gratulujeme vám. Jakmile jste vytvořili první dokument v  $\TeX$ u, je to jen otázka času, než stvoříte další a mnohem komplikovanější. Srovnajme nyní to, co jsme psali, s tím, co jsme obdrželi. Normální slova jsme psali tak, jak je to běžné, a  $\TeX$  je vytiskl běžným písmem. Avšak slovo „ $\TeX$ “, které nejde napsat z klávesnice, protože písmena nejsou všechna v řádku, jsme vložili pomocí sekvence symbolů začínající zpětným lomítkem;  $\TeX$  této sekvenci dobře porozuměl. Většina symbolů, které nejsou obyčejnými písmeny anglické abecedy, čísly nebo znaménky, se vkládá pomocí sekvencí začínajících zpětným lomítkem. Podíváte-li se trochu pečlivěji,

---

<sup>1</sup> Standardní tabulka znaků ASCII (zkratka odvozená od názvu **American Standard Code for Information Interchange**) je složena z 256 textových a řídicích znaků. Její „dolní polovina“ (0–127) se nemění, „horní polovina“ je v různých počítačích, tiskárnách, jazycích apod. využívána různě.



postřehnete, že i slovo „filologicky“ se trošku změnilo. První a druhé písmeno bylo spojeno dohromady a nad písmenem „i“ není zvláštní tečka. To je ale standardní tiskařská praxe: některá písmena se spojují dohromady a vytvářejí tzv. *ligatury*. Důvody pro tato spojení jsou estetické. Porovnejte slova „filologicky“ a „filologicky“ a všimněte si rozdílu. Poznáváme, že symbolu `\bye` ze vstupního souboru neodpovídá žádný ve výstupu. Podívejme se do stejnojmenného souboru s extenzí `log`, který TeX vytvořil při zpracování našeho pokusného souboru. Vypadá přibližně takto (v závislosti na verzi, se kterou pracujete):

```
This is emTeX, Version 3.14 [3c-beta5]
      (preloaded format=csPLAIN 93.5.29)  1 OCT 1993 23:47
**&csPLAIN mujtext.tex
(mujtext.cs [1] )
Output written on mujtext.dvi (1 page, 296 bytes).
```

Je to soubor, který bude obsahovat všechna hlášení o chybách.

▷ Cvičení 1.1 Přidejte další větu do vámi vytvořeného souboru tak, aby výsledný vstupní soubor vypadal následovně:

```
Toto je moje první filologicky nenáročná věta v \TeX u.
Byl jsem to já, kdo ji vysázel! \bye
```

Soubor zpracujte opět TeXem. Je další věta umístěna na novém řádku?

▷ Cvičení 1.2 Přidejte jako další řádek na začátek svého souboru `\nopagenumbers`. Pokuste se uhodnout, co se stane, až soubor zpracujete TeXem. Pak to vyzkoušejte a podívejte se, co se stalo.

▷ Cvičení 1.3 Přidejte ke svému souboru další tři nebo čtyři věty. Použijte v nich písmena, čísla, čárky, tečky, otazníky a vykřičníky, ale *žádné jiné další symboly*.

▷ Cvičení 1.4 Vynechte volný řádek a přidejte ještě nějaké další věty. Snadno zjistíte, že tak můžete vytvářet nové odstavce.

Seznámili jsme se se základními principy přípravy vstupních TeXových souborů. Rozvržení textu na obrazovce monitoru při vytváření vstupního souboru nemusí obecně souhlasit s formátem výsledného dokumentu. Nemůžete např. dosáhnout větších mezer mezi slovy na výstupu přidáním mezer ve vstupním souboru. Jedna nebo několik po sobě následujících mezer dá na výstupu totéž. Jak se dá očekávat, slovo na konci řádku je odděleno od dalšího na následujícím řádku mezerou. Jestliže pracujeme se souborem, který mnohokrát zpracováváme editorem, je výhodné toho využít a začínat každou novou větu na novém řádku. Mezery na začátku řádku se při zpracování TeXem samozřejmě ignorují.

▷ Cvičení 1.5 Přidejte k svému souboru následující větu jako nový odstavec a potom soubor vytiskněte.

Gratuluji! Při poslední zkoušce jste na 100% vyhověl.

Znak % se ve vstupním souboru užívá pro komentáře. Všechno, co za ním následuje až do konce řádku,  $\text{\TeX}$  prostě vynechá. Dokonce i mezera, která odděluje poslední slovo na řádku od prvního slova na dalším řádku, se ztratí. Nyní přidejte zpětné lomítko těsně před znak % a tím chybu opravte.

▷ Cvičení 1.6 Přidejte k svému souboru následující větu jako nový odstavec<sup>1</sup>:

Dlužíte mi \$10.00 a~je na čase, abyste mi je poslal!

Při zpracování tohoto vstupního souboru  $\text{\TeX}$  ohlásí chybu a zapíše ji do vašeho hlášení o chybách (je to soubor, který se jmenuje stejně jako váš, má ale extenzi log). Jestliže implementace  $\text{\TeX}$ u na vašem počítači podává hlášení o chybách při běhu a po chybě se počítač zastavil, stiskněte klávesu  $\langle \text{CR} \rangle$  a pak  $\langle \text{ENTER} \rangle$ . Výsledek se na výstupu objeví, ale je jiný, než jaký byste mohli očekávat. Prohlédněte si svůj soubor log a hledejte hlášení o chybách, o ostatní hlášení chyb se zatím nestarejte. Budeme si toho o chybách (včetně této) muset později říci mnohem víc. Nyní opravte chybu tak, že přidáte zpětné lomítko těsně před znak \$, a vše bude v pořádku. Výsledek vytiskněte.

## 1.8 $\text{\TeX}$ řídí všechno

Viděli jsme, že zpětné lomítko má speciální význam. Každé slovo, které jím začíná, bude mít při překladu vstupního souboru  $\text{\TeX}$ em zvláštní význam. Takové slovo se nazývá *řídící sekvence*, resp. *řídící posloupnost*. Existují totiž dva typy řídicích sekvencí: *řídící slovo* (např.  $\text{\TeX}$ ) je zpětné lomítko, za nímž následují pouze písmena, zatímco *řídící symbol* (např.  $\text{\$}$ ) je tvořen zpětným lomítkem, za kterým následuje jediný znak, který *není* písmenem. Protože mezera také není písmeno, je zpětné lomítko následované mezerou také přípustným řídicím symbolem. Budeme-li chtít v tomto textu zdůraznit použití mezery, použijeme zvláštního symbolu  $\sqcup$  pro tuto mezeru; tak je to např. při ukázce zvláštního řídicího symbolu  $\text{\sqcup}$ . Když  $\text{\TeX}$  čte váš vstupní soubor a narazí na zpětné lomítko, za kterým následuje písmeno, zjistí okamžitě, že se čte řídicí slovo. Proto pokračuje ve čtení písmen řídicího slova tak dlouho, až narazí na znak, který není písmenem. Jestliže tedy váš soubor obsahuje větu

Mám rád  $\text{\TeX}$ !

je řídicí slovo  $\text{\TeX}$  ukončeno vykřičníkem.

$\text{\TeX}$ book:  
7–8

---

<sup>1</sup> Adresa autora je k dispozici u překladatelů, jim můžete případně přilepšit také.

Setkáváme se ale s problémem v případě, že potřebujeme mít za řídicím slovem mezeru. Máte-li např. větu

Mám rád `\TeX` a stále ho používám.

ve svém vstupním souboru, řídicí slovo `\TeX` je ukončeno mezerou (a ta samozřejmě není písmenem). Pak ale nedostanete mezeru mezi slovy „TeX“ a „a“; vložení dalších mezer nepomůže, protože TeX mezi jednou a více po sobě následujícími mezerami nedělá rozdíl. Jestliže však za řídicí slovo napíšete řídicí symbol `\_`, uděláte současně mezeru a ukončíte i řídicí slovo. Je opravdu jednoduché zapomenout mezeru za řídicím slovem — mohou vás ujistit, že to v době, kdy se budete učit `\TeX`, uděláte nejméně jednou.

▷ Cvičení 1.7 Vytvořte vstupní soubor, který dá na výstupu odstaveček:

Mám rád `\TeX` a stále ho používám. Bude se líbit i vám. Jakmile s ním začnete zacházet, je jeho užívání opravdu jednoduché. Stačí jen zvládnout `\TeX`nické detaily.

Většina řídicích slov je zvolena tak, že jejich označení poskytuje náznak jejich použití, pokud je ovšem interpretujeme v angličtině. Můžete např. použít `\par` pro vytvoření nového odstavce (paragraph) místo použití prázdného řádku.

## 1.9 Co `\TeX` neudělá

`\TeX` je velice dobrým prostředkem pro sazbu písma, existují však ale věci, které dobře dělat neumí. Jednou z nich je tvorba obrázků. Lze vynechat místo na vložení obrázku a další starost o obrázky svěřit jiným programům. Tyto programy totiž (právě proto, že pracují s grafikou) jsou závislé na použitém počítači, zatímco `\TeX` závislý není. `\TeX` ovšem dokáže předat těmto programům jakékoli informace, vložené do souboru `dvi` pomocí příkazu `\special`. Tyto informace jsou samozřejmě závislé na použitých grafických programech.

`\TeX` sází písmo do vodorovných řádků. Jestliže má text za „základní řádek“ (baseline) cosi křivého nebo je složen z písma, jehož znaky mají proměnnou perspektivu (postupně stále větší či menší), dá se `\TeX`em zpracovávat velmi obtížně.

Viděli jsme, že cyklus „editace, `\TeX`, ovladač“ je nutno použít pro kterékoli různé formy výstupu. Je to tak dokonce i v případě, že výstupním souborem je obrazovka osobního počítače. Nelze tedy ani v tomto případě psát vstupní soubor a vidět okamžitě na obrazovce monitoru výsledek, aniž proběhne celý cyklus. Některé implementace mají k dispozici oba výstupy – jak textový, tak i grafický – současně; vidíme tedy na obrazovce velmi rychle celkový výsledek (několik sekund na stránku stačí). Spolu s poklesem cen hardwaru a vzrůstem rychlosti procesorů se můžeme dočkat dalších zlepšení.

## Kapitola 2

### Všechny velké a malé znaky

---

#### 2.1 Všechny znaky jsou významné, ale některé jsou významnější

V předcházející části jsme viděli, že většinou text vložíme z klávesnice ve formě vět složených z obyčejných slov tak, jak se to dělá na psacím stroji. Viděli jsme ještě i to, že např. zpětné lomítko se dá použít nejméně pro dva rozdílné účely. Můžeme ho použít k tomu, abychom vytiskli na výstupu něco, co nemáme na klávesnici — píšeme např. `\TeX`, abychom dostali `TeX`. Dá se také použít k tomu, abychom předali `TeXu` zvláštní pokyny: napíšeme-li `\bye`, říkáme tím, že vstupní soubor končí. Obecně každá řídicí sekvence, která musí začínat zpětným lomítkem, bude při zpracování `TeXem` chápána jako něco, co vyžaduje speciální zacházení. Existuje několik set řídicích slov, kterým `TeX` rozumí, a to ještě můžete sami definovat další — proto jsou zpětná lomítka velice důležitá. Vyžádá si to hodně času, než se postupně naučíme některá tato slova, naštěstí se jich však při běžném používání potřebuje většinou poměrně málo.

Pokud jste začínali se Spectrem, mohlo by se vám nyní začít stýskat po možnosti psát řídicí slova stiskem jedné nebo kombinace několika kláves. Časem se seznámíte i s možností, jak nahradit dlouhá řídicí slova kratšími a tak je předefinovat. Při všech těchto „trickách“ hraje prominentní roli zejména zpětné lomítko, ale nejen ono.

V principu by bylo možné nahradit všechna anglická řídicí slova českými, bylo by to však krajně neúčelné. Jedna z velkých předností `TeXu` spočívá v tom, že se stal uznávaným mezinárodním standardem. Proto je dobré ve složitějších `TeXových` dokumentech vznikajících u nás užívat k popisu angličtiny, aby je mohli využívat uživatelé `TeXu` i v zahraničí.

Existují symboly, které se jako již zmíněné zpětné lomítko užívají v `TeXu` pro speciální věci. Nyní si o všech něco řekneme. Může nás i napadnout, jak se vysází zpětné lomítko, když to budeme potřebovat. S tím vším na mysli si položíme následující otázky:

- (1) Které jsou všechny další speciální znaky?
- (2) Jak sázíme, resp. vytváříme na výstupu tyto speciální znaky v případě, že je potřebujeme použít?

`TeXbook`:  
37–38

Nemá smysl se hned začít učit z paměti k čemu se používají — zatím je potřeba vědět, že takové znaky mají speciální význam a že s nimi nelze zacházet jako s normálními písmeny. Jak se s nimi zachází bude vysvětleno později.

V tabulce jsou uvedeny všechny speciální T<sub>E</sub>Xové znaky spolu s jejich použitím a s popisem toho, jak se dají v případě potřeby vysázet (v tom případě mluvíme o doslovném (literal) výstupu):

### Znaky vyžadující speciální vstup

Znak	Použití	Vstup pro doslovný výstup
\	Pro speciální znaky/instrukce	<code>\backslash\$</code>
{	Počátek skupiny	<code>{}</code>
}	Konec skupiny	<code>}</code>
%	Komentáře	<code>%</code>
&	Tabulky a srovnávání	<code>&amp;</code>
~	Mezera, kde se nedělí	<code>~{}</code>
\$	Vstup/výstup z matematiky	<code>\$</code>
^	Exponenty v matematice	<code>^{}{}</code>
-	Indexy v matematice	<code>-{}{}</code>
#	Definice parametrů (náhrad)	<code>#</code>

## 2.2 Sazba s akcentem<sup>1</sup>

Nyní začneme používat některých T<sub>E</sub>Xových lahůdek: zatím jsme používali jen toho, že nám T<sub>E</sub>X pomáhá dostat atraktivně vypadající výstupy, ale nyní začneme s věcmi, které se např. na psacím stroji dají dělat těžko nebo se nedají udělat vůbec. Zejména si všimneme akcentů (= accents). Jak se dají vytvořit akcenty, resp. diakritická znaménka, která se na klávesnici nevyskytují? Je to stejné jako se symbolem T<sub>E</sub>X; musíme vložit do vstupního souboru určité řídicí slovo. Zdánlivě je tato partie zbytečná, protože háčky, čárky a kroužky potřebné pro psaní českých textů jsou u zčeštěné verze přístupné pohodlně prostřednictvím změny klávesnice; určitě se však někdy dostanete do situace, kdy např. v anglickém textu budete nuceni použít české jméno s akcentovanými hláskami, v českém textu německé či francouzské termíny apod.

Např. pro slovo „première“, je zapotřebí z klávesnice vložit `premi\`ere` (může se stát, že budete trochu slídit mezi klávesami, než najdete tu, na které je „levý (obrácený) apostrof“. Taková klávesa by tam někde *měla* být; T<sub>E</sub>X však zvládá i ten případ, kdy na vaší klávesnici tento znak nebo i jiné znaky chybějí). Obecně

<sup>1</sup> Jde o slovní hříčku, neboť může být chápáno jako „se (špatným) přízvukem“. Půjde ale o diakritická znaménka.

pro připojení akcentu k písmenu použijeme řídicí sekvenci (pro některý akcent je to řídicí slovo, pro jiný řídicí znak) před tímto písmenem. Uvedeme pár příkladů:

<b>TeXový vstup</b>	<b>TeXový výstup</b>
<code>\`a la mode</code>	à la mode
<code>r\`esum\`e</code>	résumé
<code>soup\c_\con</code>	soupċon
<code>No\"e1</code>	Noël
<code>na\"i_1ve</code>	naïve
<code>\v re\v richa</code>	řeřicha
<code>d\accent23um</code>	dũm

Na těchto příkladech vidíme několik věcí; většina akcentů se získá pomocí řídicích symbolů podobného tvaru. Některé se vytvářejí pomocí řídicích slov složených z jediného písmene. Někdy si to vyžádá určitou péči, protože řídicí slovo se musí ukončit např. mezerou; pokud např. ve vstupním souboru budete mít `\TeXový` místo `\TeX` ový, bude se TeX snažit nalézt řídicí slovo `\TeXový` a ne `\TeX` (u nové osmibitové verze TeXu mohou řídicí slova obsahovat i hlásky s akcenty). Za zmínku stojí (a pro češtinu je to důležité), že existuje řídicí slovo `\i`. To vytvoří „i“ bez tečky nad písmenem, tj. „ı“ (analogicky existuje i řídicí slovo s „j“ se stejným použitím) — pozor na „zrůdy“ typu „mísící automat“. V tabulkách **nepočesťujeme** označení některých akcentů (jiné názvy lze nalézt v případě potřeby v některých slovnících).

TeXbook:  
52–53

#### Akcentování bez mezer

<b>Jméno</b>	<b>TeXový vstup</b>	<b>TeXový výstup</b>
přízvuk ostrý (čárka)	<code>\`o</code>	ó
přízvuk tupý (zpětná čárka)	<code>\`o</code>	ò
přízvuk složený (stříška)	<code>\^o</code>	ô
přehláska/umlaut/dieresis	<code>\"o</code>	ö
vlnka/tilda	<code>\~o</code>	õ
pruh nad/macron	<code>\=o</code>	ō
tečka nad/dot	<code>\.o</code>	ò
kroužek nad	<code>\accent23o</code>	ô

(Zde je třeba doplnit text poznámkou: prvá tři pravopisná znaménka se užívají ve francouzštině a nazývají se *accent aigu*, *accent grave* a *accent circonflex*; písmena s přehláskou se užívají v němčině i ve francouzštině, tilda se vyskytuje mj. ve španělštině. I když lze pomocí ostrého přízvuku, háčku (viz níže) a kroužku tvořit česká písmena, je to pouze řešení z nouze. Používáme ho např. při vkládání krátkých jinojazyčných textů apod.). Vloudí-li se vám mezi sekvenci pro akcent a písmeno mezera, nic se nestane; porovnejte výsledky:  
`nap\` \i nav\` a m\` \i \v cov\`a hra`

dává napínavá míčová hra.

### Akcentování s mezerami

Jméno	$\TeX$ ový vstup	$\TeX$ ový výstup
cedilla	<code>\c o</code>	ç
tečka pod	<code>\d o</code>	ø
pruh pod	<code>\b o</code>	ö
háček	<code>\v o</code>	õ
oblouček nad/breve	<code>\u o</code>	ö
oblouček nad/tie	<code>\t {oo}</code>	öö
maďarská přehláska	<code>\H {o}</code>	ő

V tomto případě je vynechání mezery před mezi řídicím slovem pro akcent a akcentovaným písmenem závažnou chybou, která se projeví chybovým hlášením. Standardní  $\TeX$  umožňuje i sazbu písmen z mnoha jiných jazyků než z angličtiny. Uvedeme opět nějaké příklady:

### Znaky z některých neanglických jazyků

$\TeX$ ový vstup	$\TeX$ ový výstup	Příklad použití
<code>\AE, \ae</code>	Æ, æ	Ægean, æsthetics
<code>\OE \oe</code>	Œ, œ	Œuvres, hors d'œuvre
<code>\AA, \aa</code>	Å, å	Ångstrom
<code>\O, \o</code>	Ø, ø	Øre, København
<code>\L, \l</code>	Ł, ł	Łódz, łódka
<code>\ss</code>	ß	nuß
<code>!~</code>	¡	¡HOLA!
<code>?~</code>	¿	¿Con qué sueña?
<code>{\it\}\$}</code>	£	1 £ = 52 Kč

Vysázejte všechny příklady z následujících cvičení:

- ▷ Cvičení 2.1 Rozumí Æschylus Œdipovi?
- ▷ Cvičení 2.2 Nejmenší vnitřní jednotka délky v  $\TeX$ u je asi 53.63Å.
- ▷ Cvičení 2.3 Élèves,<sup>1</sup> réfusez vos leçons! Jetez vos chaînes!
- ▷ Cvičení 2.4 Zašto tako polako pijete čaj?
- ▷ Cvičení 2.5 Ich muß heißen Tee abkühlen.

<sup>1</sup> Francouzi ovšem akcenty nad velkými písmeny ve skutečnosti nepíší.

- ▷ Cvičení 2.6 Maar die ys is lekker.
- ▷ Cvičení 2.7 Peut-être il préfère le café glacé.
- ▷ Cvičení 2.8 Lze jet trajektem z Ölandu do Ålandu?

## 2.3 Tečky, pomlčky, uvozovky, . . .

Psaní na stroji bylo odjakživa kompromisem; malé množství kláves ve srovnání s množstvím potřebných symbolů si vynucovalo při psaní zpravidla změny. Když však text připravujeme pomocí  $\text{\TeX}$ u, nemusíme se nijak omezovat. V této části se seznámíme s některými rozdíly mezi psaním na stroji a používáním  $\text{\TeX}$ u.

Existují čtyři druhy pomlček, které se běžně používají. Krátká pomlčka, resp. spojovník či rozdělovník (hyphen) se užívá ke spojování (je-li, Hahn-Banach, . . .) nebo k dělení slov na konci řádku. Pomlčka (en-dash, kde „en“ souvisí se šířkou písmene  $n$ ) se užívá při výčtech (např.: str. 54–58), dlouhá pomlčka (em-dash, kde „em“ indikuje šířku písmene  $M$ ) je v podstatě gramatickým symbolem užívaným někdy pro — jsou-li nezbytně nutné — vsuvky, atp. Konečně užíváme znamení „minus“ v matematických textech; příklady udává tabulka.

$\text{\TeX}$ book:  
3–5

### Různé typy pomlček

Jméno	$\text{\TeX}$ ový vstup	$\text{\TeX}$ ový výstup	Příklad
rozdělovník	-	-	Prostor je 3-rozměrný.
pomlka	--	-	Viz. stránky 3–4.
dlouhá pomlka	---	—	Vidím je — jsou živí!
minus	\$- $\$$	-	Teplota byla -3 stupně.

(Poznámka: Zde je třeba se při sazbě přidršet norem pro tisk, a ne norem pro psaní strojem, kde se — vynuceně — všechny pomlčky píší stejně. Na rozdíl od angličtiny necháváme ale mezeru po obou stranách „—“.) Dlouhá pomlka „—“ se u nás běžně užívala v klasické sazbě knižtiskové, i když samozřejmě její délka není ve všech písmech stejná. S příchodem fotosazby se ovšem začaly vytrácet i běžné ligatury fi, ff, fl, které většina moderních sazbecích systémů už vůbec nezná. Podobně většina uživatelů ventur a pagemakerů vystačí s rozdělovníkem místo pomlčky, jak se můžeme denně přesvědčovat v řadě našich deníků.  $\text{\TeX}$  je od verze 3.0 mimo jiné schopen v jednom řezu písma pracovat s více než 30 000 ligaturami, přičemž se nemusí jednat o jednoduché záměny dvojic znaků jiným znakem. Ligaturní mechanismus  $\text{\TeX}$ u se např. využívá v azbukových písmech  $\mathcal{A}\mathcal{M}\mathcal{S}$ - $\text{\TeX}$ u, kde po použití obvyklé mezinárodní transkripce dostanete na výstupu správný text v azbuce.

U každé ukázky ve cvičeních z této sekce napište text, který je nutno vložit do vstupního souboru, aby výsledný text na výstupu byl stejný jako text ve cvičení.



▷ Cvičení 2.9 Zpívala-li tra-la-la, ráda k tomu — poněkud neobratně — poskakovala.

▷ Cvičení 2.10 Zima 1942–1943 byla nejhorší — mnoho lidí nepřežilo.

Další rozdíl mezi psaním na stroji a psaním pomocí T<sub>E</sub>Xu je v užívání uvozovek. T<sub>E</sub>X je vytváří z levých a pravých apostrofů: na začátku použijeme ``, na konci ´´, což by dalo bez dalšího zásahu uvozovky dle amerického stylu (obojí nahoře). Stejně lze psát i jednoduché uvozovky, které se u nás neužívají. V českých textech se užije makro `czech.sty`, které upraví “anglosaské” uvozovky na „české“ (též upraví mezerování mezi větami a další věci). Užijeme-li normální uvozovky <> z klávesnice, dostaneme zpravidla uvozovky koncové, tj. ”. Jejich výlučné užívání nedoporučujeme; nelze na to ostatně zcela spoléhat, někdy to tak nevyjde. Pokud jste vynechali kapitolu o zčeštěném T<sub>E</sub>Xu, vraťte se k ní, seznámíte se s dalšími možnostmi.

▷ Cvičení 2.11 „Jeho jasnost,“ ohlásil komoří.

▷ Cvičení 2.12 Jaroslav vyhrkl: „Opravdu je taková, že neumí říci ‚ne‘?“

Někdy užíváme skupinu tří teček k označení místa, kde došlo k vypuštění části textu, nebo pro označení běhu času apod. Tři tečky vložené z klávesnice dají na výstupu též tři tečky, ale velmi blízko u sebe. Trojici teček s „dobrymi“ mezerami dostaneme v tomto případě pomocí řídicího slova `\dots`, které vložíme do vstupního souboru.

T<sub>E</sub>Xbook:  
173

▷ Cvičení 2.13 Jistě si vzpomenete na heslo, které na nás hlásalo z poutačů „... až na věky ... a nikdy jinak.“

V anglickém textu (zejména v americké angličtině) je mezeru za tečkou na konci věty větší než mezeru mezi slovy (při psaní českého textu se pomocí souboru `czech.sty` automaticky vkládá řídicí slovo `\frenchspacing`, které tuto zvláštnost potlačí). Proto je třeba T<sub>E</sub>X informovat o tom, že například tečka za zkratkou, která končí malým písmenem, *není* na konci věty. To lze udělat dvěma způsoby: za tečkou lze napsat `\_` nebo `~`, čímž se mezerování změní. Druhá varianta dává mezeru, v níž nelze dělit řádek; je např. použitelná pro vstup `Prof.~Knuth`. Je vhodné ji užívat s tituly, se zkratkami křestních jmen, atp. Všimněte si, že se v tomto případě v kombinaci s `~` neužívá zpětného lomítka.

T<sub>E</sub>Xbook:  
91–92

▷ Cvičení 2.14 Mr. J. B. Smith z Washingtonu, D. C., přijede z U. S. A. do ČSFR<sup>1</sup> v pátek 13. 8. v 9.30.

---

<sup>1</sup> V případě změny názvu nebo neexistence vhodně upravte.

▷ Cvičení 2.15 Nezkracujte pravou délku úsečky následovně: pr. d. úsečky; raději pište pr. dél. úsečky nebo ještě lépe pr. délka úsečky. (Prof. K. Havlíček.)

## 2.4 Různá písma

Nejvíce patrný rozdíl mezi strojopisem a produktem  $\TeX$ u je bezpochyby v různorodosti typů písma (písmena jednoho typu tvoří písmo, nebo-li font) a různých symbolů. Když  $\TeX$  začíná pracovat, má k dispozici šestnáct písem — jejich úplný seznam je obsažen v Appendixu F Knuthovy knihy  $\TeX$ book. Většina z nich se užívá automaticky; v matematickém textu se index či exponent tiskne menším typem písma bez dalšího zásahu uživatele.

$\TeX$ book:  
427–432

Následující tabulka podává přehled nejčastěji užívaných písem (ukázkou a některé názvy nepřekládáme):

### Ukázky písem

Jméno písma	$\TeX$ ový přepínač	Vzorek písma
Antikva/Roman	<code>\rm</code>	This is roman type.
Polotučně /Boldface	<code>\bf</code>	<b>This is boldface type.</b> <sup>1</sup>
Kurzíva/Italic	<code>\it</code>	<i>This is italic type.</i>
Skloněné/Slanted	<code>\sl</code>	<i>This is slanted type.</i>
Stroj/Typewriter	<code>\tt</code>	This is typewriter type.
Matematické symboly	<code>\cal</code>	<i>SCRIPT LETTERS.</i> <sup>2</sup>

Chceme-li změnit obvyklé stojaté písmo (u nás se zpravidla nazývá *antikva*, roman v anglických textech) např. na *kurzívu* (italic), použijeme řídicího slova `\it`. Zpět k antikvě se vrátíme pomocí `\rm`. Chceme-li například napsat větu “Začneme s antikvou, *teď přejdeme na kurzívu* a nyní opět k antikvě”, stačí k tomu napsat pro vstup ‘‘Začneme s antikvou, `\it` teď přejdeme na kurzívu`\` `\rm` a nyní opět k antikvě’’.<sup>3</sup>

Šikmé písmo (slanted) a kurzíva se obvykle na první pohled výrazně neliší (porovnejte ale šikmé *a* s kurzívním *a*). Když se mění typ písma ze šikmého nebo z kurzívy na stojatou antikvu (či jiné podobné písmo), poslední šikmé písmeno se naklání k prvnímu stojatému. Nevypadá to hezky a je potřeba zkrácení mezery

<sup>1</sup> Významový posuv u překladu lépe odpovídá československým zvyklostem.

<sup>2</sup> Tento příklad je drobný podvod, protože pro jeho vytvoření je třeba něco vědět o vkládání matematických textů, chtěl jsem ho však stejně uvést. Až se seznámíte s používáním matematických symbolů, budete umět použít i toto písmo. Konečně ani vkládání poznámek pod čarou do tabulek není tak docela jednoduché.

<sup>3</sup> Záleží na použitém zobrazovači, jestli se vám na monitoru objeví `ŧ`, `đ`, anebo `ť`, `ď` — u zobrazovače pro VGA dodávaného  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{U}\mathcal{G}\mathcal{E}\mathcal{M}$  nastane ten druhý případ.

$\TeX$ book:  
7–8

kompensovat přidáním další menší mezery pro přesah — nazývá se to *kurzívní korekce* (italic correction). Provádí se pomocí řídicího symbolu `\/`. V následující větě pro srovnání *není* po prvním použití kurzívy užito kurzívní korekce, ale po druhém ano: *Pif* koupil koně a *Paf* koupil sedlo. Podíváte-li se pozorněji, jsou v prvním případě písmena „f“ a „k“ příliš blízko sebe; s korekcí je sazba hezčí.

V  $\TeX$ u se dají definovat nová písma samozřejmě jen tehdy, jsou-li ve vašem systému instalována. Různé velikosti písma se dají užívat pomocí řídicího slova `\magstep`. Pro definici nového písma je třeba znát jeho označení ve vašem počítači. Například obvyklé stojaté písmo (roman) se ve většině systémů i na osobních počítačích jmenuje „cmr10“. Máte-li ve vstupním souboru řádek `\font\bigrm = cmr10 scaled \magstep 1`, můžete pak používat řídicí slovo `\bigrm` stejným způsobem jako užíváte `\it` a `\rm`.

$\TeX$ book:  
13–17

Přepnutím pomocí `\bigrm` dostanete antikvu zvětšenou o 20 % oproti normální. Definujete-li `\font\bigbigrm = cmr10 scaled \magstep 2`, dostanete písmo o 44 % větší než normální antikva. Dají se použít velikosti od `\magstep 0` až po `\magstep 5`. Na většině počítačů je k dispozici i `\magstephalf`; pomocí tohoto řídicího slova dostaneme zvětšení o 9,5 %. Uveďme si vzorky téhož písma v různých velikostech:

Toto je vzorek s magstep 0, což je obvyklá velikost.

Toto je vzorek s magstep 1.

Toto je vzorek s magstep 2.

Toto je vzorek s magstep 3.

Toto je vzorek s magstep 4.

Toto je vzorek s magstep 5.

Je dokonce možné definovat zcela nová písma. Závisí to ovšem opět na tom, zda jsou v systému instalována (zde nejde o tvorbu zcela nových písem pomocí speciálního softwaru, ale o užití již hotových písem, kterých se v závislosti na zakoupeném programovém vybavení vyskytuje mnoho). Tak např. v mnoha instalacích lze nalézt soubor označený `cmss10`, což je označení pro bezpatkové písmo (sans serif). Použijeme-li definice písma `\font\sff = cmss10`, lze pak psát řídicí slovo `\sf` k přepnutí do tohoto písma stejně jako u `\bf`, které přepíná do polotučného písma (boldface). Jestliže použijeme zmíněnou definici, pak vložením

`\sf` Toto je ukázka našeho bezpatkového (Sans Serif) písma.

dostaneme

Toto je ukázka našeho bezpatkového (Sans Serif) písma.

Existuje mnoho dalších použitelných typů písem. Písma uvedená v následující tabulce patří k nejběžnějším.

### Názvy externích druhů písma<sup>1</sup>

CMBSY10 *	CMBXSL10	CMBXTI10	CMBX10	CMBX12	CMBX5
CMBX6	CMBX7	CMBX8	CMBX9	CMB10	CMCSC10
CMDUNH10	CMEX10 *	CMFF10	CMFIB8	CMFI10	CMITT10
CMMIB10 *	CMMI10 *	CMMI12 *	CMMI5 *	CMMI6 *	CMMI7 *
CMMI8 *	CMMI9 *	CMR10	CMR12	CMR17	CMR5
CMR6	CMR7	CMR8	CMR9	CMSLTT10	CMSL10
CMSL12	CMSL8	CMSL9	CMSLBX10	CMSSDC10	CMSSI10
CMSSI12	CMSSI17	CMSSI8	CMSSI9	CMSSQI8	CMSSQ8
CMSS10	CMSS12	CMSS17	CMSS8	CMSS9	CMSY10 *
CMSY5 *	CMSY6 *	CMSY7 *	CMSY8 *	CMSY9 *	CMTSC10
CMTEX10 *	CMTEX8 *	CMTEX9 *	CMTI10	CMTI12	CMTI7
CMTI8	CMTI9	CMTT10	CMTT12	CMTT8	CMTT9
CMU10	CMVTT10				

Řekněme si něco o těchto názvech. Prvá dvě písmena CM znamenají Computer Modern; tento název pro celou rodinu písem zavedl jejich tvůrce D. Knuth. Číslo na konci udává velikost písma v bodech: 10 bodové písmo má obvyklou velikost, 7 bodová velikost je běžná pro indexy a 5 bodová velikost je velikost písma pro indexy druhé úrovně; písmo 12 bodové je o 20 % větší než 10 bodové atd. Následují-li po písmenech CM písmeno B (od boldface), jde o polotučné písmo. Podobně R indikuje antikvu (roman), I kurzívu (italic). Další běžné označení CSC se užívá pro kapitálky (nesouvisí s hodnotí CSc., ale s názvem „small caps“), SL pro skloněné písmo (slanted), SS pro bezpatkové písmo (sans serif), SY pro symboly a TT pro písmo typu psacího stroje.

▷ Cvičení 2.16 Nalezněte všechny druhy písma, které máte na počítači, a v několika z nich vytiskněte všechna písmena a číslice.

▷ Cvičení 2.17 Písmo CMR12 je o 20 % větší než CMR10. Také `\magstep 1` zvětšuje písmo o 20 %. Vezměte si nějaký text a vytiskněte jej pomocí CMR12 a potom pomocí CMR10 zvětšeného pomocí `\magstep 1`. Výsledky se zcela liší!

<sup>1</sup> V  $\text{\LaTeX}$  je nutné při použití přímých akcentů zavést místo fontu `cm*` font `cs*` (tedy například místo fontu `cmr10` píšete `csr10`). Vyjimku tvoří fonty matematické (v tabulce označené hvězdičkou), pro něž alternativa `cs*` neexistuje.

## Kapitola 3

### Úprava věcí příštích

---

V této části chceme ukázat, jak vytvořit *text* různých tvarů či velikostí. Budeme tak moci tvůrčím způsobem mnohem více ovládat tvar vytvářeného dokumentu. Lze však samozřejmě používat  $\text{\TeX}$  s již předdefinovanými (default) velikostmi tak, jak jsme to dělali dosud. Popisujeme-li velikosti různých částí stránky textu, můžeme užívat několika různých délkových jednotek.

#### 3.1 Jednotky, jednotky, jednotky

$\text{\TeX}$  může užívat mnoho typů délkových jednotek. Nejobvyklejší jsou centimetr, palec (inch), tiskařský bod a pica. Užívané zkratky dle uvedeného pořadí jsou `cm`, `in`, `pt` a `pc`. (Nejsou to všechny jednotky, které  $\text{\TeX}$  zná, stejně tak rozumí i zkratce `mm`, nicméně nejbližší jsou mu palce — předdefinované rozměry jsou zpravidla udávány v `in`.) Bod (tiskařský dále vynecháváme) je definován rovností  $1 \text{ in} = 72,27 \text{ pt}$ , pica se definuje pomocí vztahu  $1 \text{ pc} = 12 \text{ pt}$ . Bod je tedy dosti malá míra — následující tabulka dává rozumnou představu o uvedených mírách. V evropské typografii se ovšem používají trochu odlišné míry: 1 bod (Didotův, `dd`) je o malinko větší než americký bod (je  $1 \text{ dd} = \frac{1238}{1157} \text{ pt}$ ) a cicero (`cc`) je definováno podobně jako pica vztahem  $1 \text{ cc} = 12 \text{ dd}$ .

$\text{\TeX}$ book:  
57

1 palec (in) : \_\_\_\_\_  
1 centimetr (cm) : \_\_\_\_\_  
20 bodů (pt) : \_\_\_\_\_  
20 bodů (dd) : \_\_\_\_\_  
1 pica (pc) : \_\_\_\_\_  
1 pica (cc) : \_\_\_\_\_  
1 milimetr (mm) : u

Bodů se tedy používá k provádění jemných změn; pica je cca vzdálenost mezi základnami (jde o „linky“, na něž se písmena umísťují, i když některá písmena jako *p*, *g* apod. tuto linku směrem dolů přesahují) dvou po sobě následujících řádků (nezvětšeného) normálního textu.  $\text{\TeX}$  je velmi přesný; jeho nejmenší vnitřní jednotka délky je menší než čtyřmilióntina palce (mimoходом: umíte si představit, s jakými rezervami vzhledem k budoucímu vývoji je  $\text{\TeX}$  koncipován?). S ohledem na tento fakt je to pouze rozlišovací schopnost výstupní tiskárny, která určuje možnou přesnost výstupu.

Někdy je užitečné používat ještě další dvě jednotky, jejichž velikost *se mění* s velikostí užívaného písma, kterým v daném okamžiku píšeme. Jednotka `ex` je cca výška malého „x“ a `em` je trošku menší než šířka velkého „M“.

TeXbook:  
60

Forma výstupu je obecně určena užitými řídicími slovy. Těchto slov je spousta, proto je možné výstup v tištěné formě ovládat velice jemným způsobem. Zároveň je však zřejmé, že pro běžnou práci jich většinou stačí pouze malé množství.

## 3.2 Úprava stránky

Stránka textu má tři základní části. Nad hlavním textem bývá obvykle *záhlaví* (headline). Záhlaví zpravidla obsahuje např. název kapitoly, název jejího oddílu (section) nebo číslo stránky; může se lišit u levých a pravých stránek. Pod záhlavím je hlavní text, který obsahuje i poznámky (někdy mluvíme o „poznámkách pod čarou“). Pod ním následuje poslední část stránky (footline), obsahující eventuálně (ne nutně) číslo stránky apod.

V příkladech, s nimiž jsme se až dosud seznámili, bylo záhlaví prázdné; také třetí část strany kromě čísla stránky uprostřed neobsahovala nic — v případě, že jsme užili řídicího slova `\nopagenumbers`, potlačujícího tisk čísel stránek, byla i tato část prázdná. O první a poslední části stránky bude později ještě řeč, teď se však soustředíme na hlavní text.

Pro přechod na novou stránku (z jakékoli pozice na stránce) můžeme použít řídicích slov `\vfill` `\eject`. Řídicí slovo `\eject` si vynucuje přechod na další stranu, zatímco `\vfill` vyplní prázdným místem zbytek aktuální strany na jejím spodním konci (můžete zkusit pokusně vynechat `\vfill`; uvidíte, jak pracuje vertikální obdoba zarovnávání pravého okraje řádků).

Aktuální užitá vodorovná šířka textu na stránce je popsána pomocí řídicího slova `\hsize`. Dá se ale kdykoli změnit např. na 4 palce použitím řídicího slova `\hsize = 4 in`. Jak lze vidět na tomto odstavci šířka textu se dá rovněž změnit (v tomto případě také na 4 palce) i v rámci jediného odstavce. Protože tato označuje novou šířku odstavce ve vodorovném směru, dává výraz typu `\hsize = .75\hsize` změnu na novou hodnotu vztaženou k předcházející, v našem případě od 4 in na 3 in.

Svislá (vertikální) obdoba řídicího slova `\hsize` je `\vsize`. Přiřazenou hodnotu lze měnit stejně jako v předcházejícím případě. Tak např. `\vsize = 8 in` (ale i `\vsize=8in` nebo `\vsize8in`) lze užít ke změně svislého rozměru hlavního textu. Připomínáme, že `\vsize` je rozměr textu *kromě* záhlaví a místa pod patou stránky, kde je obecně místo pro footline. Text se dá též po stránce posunovat. Horní levý roh textu je vzdálen o 1 in ( $\doteq 25$  mm) od levého a horního okraje papíru. Řídicí slovo `\hoffset` a `\voffset` slouží k posunutí textu ve vodorovném, resp. svislém směru. Tak např. `\hoffset = 10mm` a `\voffset = -.2 in` posunou text o dalších

10 mm vpravo a cca 5 mm nahoru; můžeme užívat najednou více jednotek,  $\TeX$  si je lehce přepočítá.

### Řídicí slova pro rozměry stránky

Jméno	Řídicí slovo v $\TeX$ u	v $\TeX$ u předdefinováno
šířka textu	<code>\hspace</code>	6,5 in
výška textu	<code>\vspace</code>	8,9 in
levý okraj	<code>\hoffset</code>	0 in
horní okraj	<code>\voffset</code>	0 in

Připomínáme, že text je automaticky orientován vzhledem k bodu, který je vzdálen 1 in dolů a 1 in doprava od levého horního rohu stránky; např. vložení instrukce `\hoffset = 5 mm` způsobí, že text bude umístěn asi 30 mm od levého okraje stránky, `\hoffset = -5 mm` posune text asi na 20 mm od levého okraje stránky. Zde si uvědomte, že při psaní českého textu zpravidla rozměr stránky měníme, např. použitím `\input plaina4`. (Tato změna je již v  $\LaTeX$ u v zahrnuta do formátu `cspain`).

▷ Cvičení 3.1 Napište jeden odstavec textu a pak ho několikrát za sebou okopírujte. Vložte na začátek `\hspace = 5 in` a za první odstavec `\hspace = 10 cm`. Zkuste totéž s jinými hodnotami pro `\hspace`.

▷ Cvičení 3.2 Vložte do vytvořeného souboru z předešlého cvičení na začátek `\hoffset = .5 in` a `\voffset = 1 in`. Zkuste `\hoffset.5in` a `\voffset1in`.

▷ Cvičení 3.3 Vložte ještě do souboru z předešlého cvičení před první odstavec `\vspace = 2 in` a znovu ho zpracujte.

V předešlé části jsme viděli, že lze použít písma s větší velikostí pomocí řídicího slova `\magstep`. Dá se však zvětšit celý dokument najednou? Položíte-li `\magnification = \magstep 1` na začátek souboru, pak se celý váš dokument zvětší asi o 20%. Jiné hodnoty pro `\magstep` jsou také přípustné. **Pozor, `\magnification` se dá užít pouze před vytištěním třeba jen jediného písme!** Zvětšování přináší s sebou problém jednotek. Jestliže je ve vstupním souboru předepsáno `\hspace = 5 in`, je při 20% zvětšení nový rozměr 6 palců? Pokud nepodnikneme nic dalšího, je to tak: `\hspace` bude při výstupu rovno 6 palcům; všechny rozměry se stejnoměrně zvětší, užijeme-li `\magnification`. Jsou ale speciální situace, kdy je nežádoucí, aby něco podobného nastalo: potřebujete například vynechat přesně 10 cm pro obrázek. V takovém případě lze kteroukoli jednotku modifikovat pomocí `true`. Tak např. `\hspace = 5 true in` dá za všech okolností délku řádků 5 palců, bez ohledu na použité zvětšení.

▷ Cvičení 3.4 Vložte `\magnification = \magstep 1` na první řádek do některého souboru z minulých cvičení a srovnajte, co se změní na výstupu.

### 3.3 Úprava odstavce

Když  $\text{T}_{\text{E}}\text{X}$  čte ze vstupního souboru `text`, který má být vytištěn, čte jej po odstavcích. Prakticky to znamená, že lze značně ovlivňovat tvar odstavce, ale je zapotřebí určité opatrnosti. Již jsme viděli, že `\hspace` se dá použít k řízení šířky odstavce. Předpokládejme ale, že ve vašem souboru by bylo použito

```
\hspace = 5 in
Za devatero horami a~devatero řekami...
$\vdots$
... byla krásná jako obrázek.
\hspace = 6.5 in
```

Jaká bude výsledná šířka odstavce? Parametr `\hspace` byl změněn na začátku odstavce a potom ještě na konci. Protože nebyl odstavec ještě dokončen (užitím prázdného řádku nebo `\par`), bude vytištěn se šířkou 6,5 palce. Kdyby mezi oba údaje o `\hspace` byl vložen např. prázdný řádek, pak by byl odstavec až k tomuto řádku vytištěn se šířkou 5 palců. Obecně při tvorbě odstavce hodnoty parametrů těsně před jeho dokončením určují jeho skutečný tvar.

Následující tabulka obsahuje některé parametry, kterými se ovládá tvar odstavce:

Některé parametry tvaru odstavce

Funkce	Řídicí slovo v $\text{T}_{\text{E}}\text{X}$ u	Předdefinováno
šířka odstavce	<code>\hspace</code>	6,5 in
odsazení prvního řádku odstavce	<code>\parindent</code>	20 pt
vzdálenost mezi řádky	<code>\baselineskip</code>	12 pt
vzdálenost mezi odstavci	<code>\parskip</code>	0 pt

Řídicí slovo `\noindent` lze použít na začátku odstavce pro potlačení dříve definovaného (resp. předdefinovaného) odsazení v prvním řádku odstavce. Taková úprava ovlivní právě jeden odstavec; je samozřejmé, že stejného efektu docílíme pomocí `\parindent = 0 pt` pro všechny následující odstavce.

Řídicí slovo `\vskip` lze použít k vložení místa mezi odstavce. Když mezi dva odstavce vložíme `\vskip 1 in`, vytvoříme volný prostor o výšce 1 palec. **Pozor, prostor není vložen, přecházíme-li mezi odstavci na novou stránku!** Objeví-li se `\vskip 1 in` na začátku nové stránky, bude výsledek stejný, jako kdyby toto řídicí slovo bylo vypuštěno. Pro řadu situací, kdy pracujeme s volným



místem, je to vcelku výhodné. Např. volné místo před začátkem nové části, sekce, odstavce apod. by nemělo začínat, ani pokračovat na nové stránce.

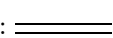
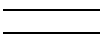
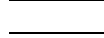
TeXbook:  
352

Jak to ale zařídíme, chceme-li opravdu mít volné místo na začátku stránky např. u titulní strany či u první strany kapitoly? Stránku lze začít např. řídicím symbolem `\_`, ale pak je ve skutečnosti na začátku strany prázdný řádek. Pokud není dále něco tištěno, přidá použití řídicích slov `\baselineskip` a `\parskip`, další volný prostor. Nejjednodušší je použít místo `\vskip` řídicí slovo `\vglue`, které bude mít potřebný účinek. Tak např. `\vglue 1 in` vynechá volné místo o výšce 1 palec i na začátku stránky.

TeXbook:  
115

Jiná metoda spočívá v použití řídicích slov `\topinsert` a `\endinsert`. Je-li použita konstrukce `\topinsert ... \endinsert`, pak se text mezi `\topinsert` a `\endinsert` objeví na stránce nahoře. V tomto případě se `\vskip` nepotlačí, tj. vynechání se provede. Je to zejména užitečné při vynechávání místa pro obrázky.

Existují i určitá řídicí slova, která vytvářejí volný prostor o předepsané (malé) výšce. Jsou to `\smallskip`, `\medskip` a `\bigskip`. Jejich velikosti se dají lehce znázornit vzdálenostmi vodorovných linek v následujícím schématu:

`\smallskip`:  `\medskip`:  `\bigskip`: 

Pružnější řízení šířky odstavce nebo celého textu je možné pomocí řídicích slov `\rightskip` a `\leftskip`. Položíme-li `\leftskip = 20 pt`, pak bude levý okraj odstavce posunut o dalších dvacet bodů doprava. Lze použít i záporné hodnoty pro `\leftskip`; ty způsobí posun levého okraje odstavce „ven“ doleva. Řídicí slovo `\rightskip` způsobuje obdobné úpravy na pravém okraji odstavce. Uvedeme-li tyto instrukce kdekoliv v průběhu psaní odstavce, provedou se od začátku tohoto odstavce. Také každý následující odstavec bude sázen s těmito hodnotami pro `\leftskip` a `\rightskip`. Chceme-li takto upravit pouze jeden odstavec, je nutno v dalším odstavci použít stejnou konstrukci s opačnou hodnotou nebo využít možnosti „lokalizace“ změny, o které se dočtete v následující kapitole. Řídicí slovo `\narrower` je ekvivalentní současnému použití obou slov `\leftskip` a `\rightskip` s hodnotou, která je rovna aktuální hodnotě `\parindent`. Je to velmi výhodné např. pro dlouhé citace — tento odstavec je příkladem využití řídicího slova `\narrower`. Dvojití použití za sebou zúží odstavec o dvojnásobek aktuální hodnoty `\parindent`. Změna platí i v následujících odstavcích; o lokálním použití se dočtete dále.

TeXbook:  
100

▷ Cvičení 3.5 Vytvořte dva odstavce podle následujících instrukcí: levý okraj obou odstavců je posunut vpravo o 1,5 palce, pravý je posunut vlevo o 0,75 palce a mezi oběma odstavci je vynechán prostor právě o výšce 1 palec.

I v rámci jednoho odstavce lze měnit jeho tvar pomocí jednoho z řídicích slov `\hangindent` anebo `\hangafter`. Velikost zkrácení řádků je řízena `\hangindent`; pokud je přiřazená hodnota kladná, řádky se zkracují zleva, a pokud je záporná, řádky se zkracují zprava („vykousnuté místo“ je tedy buď vlevo, nebo vpravo). O tom, které řádky budou kratší, rozhoduje hodnota proměnné `\hangafter` na začátku odstavce. Je-li `\hangafter` kladné (celé) číslo, potom určuje počet řádků plné (normální) délky. Jsou-li `\hangindent = 1.75 in` a `\hangafter = 6` parametry pro tento odstavec, pak prvních šest řádků bude mít plnou délku a zbývající řádky odstavce budou zleva zkráceny o 1,75 palce. Zvolíme-li `\hangindent = -1.75 in` a `\hangafter = -6`, potom prvních šest řádků bude zkráceno zprava o 1,75 palce a zbývající budou mít plnou délku. Po každém odstavci  $\TeX$  automaticky nastavuje předdefinované hodnoty `\hangindent=0 pt` a `\hangafter = 1`. Tato řídicí slova se hodí pro „odskoky v rámci odstavce“, resp. pro obtečení prázdného místa (např. pro obrázek) odstavcem. Řídicí slovo `\hang` na začátku odstavce dosadí `\hangafter=1`; proto bude mít první řádek odstavce obvyklou délku a další se zkrátí o hodnotu `\parindent`, tj. `\hangindent` se položí roven hodnotě `\parindent`.

$\TeX$ book:  
355

$\TeX$ book:  
102

Zopakujeme text právě přečteného odstavce, přičemž položíme `\hangafter = -6` a `\hangindent = -1.75`.

I v rámci jednoho odstavce lze měnit jeho tvar pomocí řídicích slov `\hangindent` a `\hangafter`. Velikost zkrácení řádků je ovládána `\hangindent`; pokud je přiřazená hodnota kladná, řádky se zkracují zleva a pokud je záporná, řádky se zkracují zprava („vykousnuté místo“ je tedy buď vlevo, nebo vpravo). O tom, které řádky budou kratší, rozhoduje hodnota proměnné `\hangafter` na začátku odstavce. Je-li `\hangafter` kladné (celé) číslo, potom určuje počet řádků plné (normální) délky. Jsou-li `\hangindent = 1.75 in` a `\hangafter = 6` parametry pro tento odstavec, pak prvních šest řádků bude mít plnou délku a zbývající řádky odstavce budou zleva zkráceny o 1,75 palce. Jestliže zvolíme `\hangindent = -1.75 in` a `\hangafter = -6`, potom prvních šest řádků bude zkráceno zprava o 1,75 palce a zbývající budou mít plnou délku. Po každém odstavci si  $\TeX$  automaticky nastavuje předdefinované hodnoty `\hangindent = 0 pt` a `\hangafter = 1`. Tato řídicí slova se hodí pro „odskoky v rámci odstavce“, resp. pro obtečení prázdného místa (např. pro obrázek) odstavcem. Řídicí slovo `\hang` na začátku odstavce dosadí `\hangafter = 1`; proto bude mít první řádek odstavce obvyklou délku a další se zkrátí o `\parindent`, tj. `\hangindent` se položí roven hodnotě `\parindent`.

$\TeX$ book:  
355

$\TeX$ book:  
102

$\TeX$ book:  
101

K vytvoření odstavců rozmanitých tvarů lze užít řídicí slovo `\parshape`.

Další užitečné řídicí slovo k úpravě odstavců je `\item`. Používá se dle vzoru `\item{...}`. Vytváří se jím odstavec, v němž je každý řádek zkrácen o `\parindent`; první řádek je označen zleva tím, co jsme dali mezi složené závorky. Obvykle se ho užívá v kombinaci s řídicím slovem `\parskip = 0 pt`, abychom nedostali příliš velké mezery mezi řádky. Řídicí slovo `\itemitem` je stejné jako `\item`, jediný rozdíl je ve zkrácení řádků o dvojnásobek hodnoty `\parindent`. Vše nejlépe vysvětlí následující příklad: po vložení do vstupního souboru obdržíme z následující konstrukce

```
\parskip = 0 pt \parindent = 30 pt \noindent
Vyřešte následující úkoly:
\item {(1)} Připravte se a~maximálně se soustředte.
Cesta k~řešení úkolů je někdy namáhavá a~svízelná. Najděte
nejprve odpověď na otázku (2)!
\item {(2)} Najděte odpověď na otázku (3)!
\item {(3)} Najděte odpověď na otázku (3a)!
\itemitem {(3a)} Nejste-li unaveni, pokračujte až do
úplného vyřešení všech úkolů. Nyní najděte odpověď na otázku (3b)!
\itemitem {(3b)} Najděte odpověď na otázku (1)!
```

tento text

Vyřešte následující úkoly:

- (1) Připravte se a maximálně se soustředte. Cesta k řešení úkolů je někdy namáhavá a svízelná. Najděte nejprve odpověď na otázku (2)!
- (2) Najděte odpověď na otázku (3)!
- (3) Najděte odpověď na otázku (3a)!
  - (3a) Nejste-li unaveni, pokračujte až do úplného vyřešení všech úkolů. Nyní najděte odpověď na otázku (3b)!
  - (3b) Najděte odpověď na otázku (1)!

### 3.4 Úprava řádku

Většinou TeX při lámání odstavce pracuje spolehlivě. Někdy je však nutné přidat další instrukce. Vynutit si přechod na nový řádek je možné vložением `\hfil` `\break` do vstupního souboru. V případě potřeby lze umístit určitý text na samostatný řádek pomocí řídicího slova `\line{...}`; pak se text umístěný mezi složené závorky vytiskne na jediný řádek (i když výsledek může být někdy hrozný). Řídicí slova `\leftline{...}`, `\rightline{...}`, a `\centerline{...}` umístí text v závorce na jeden řádek. Přitom text bude v prvním případě umístěn k levému okraji řádku, ve druhém k pravému a ve třetím bude umístěn doprostřed řádku. Tedy následující text ve vstupním souboru

```
\leftline{Teď jsem vlevo nahoře.}
\centerline{Teď jsem uprostřed.}
\rightline{Teď jsem vpravo.}
```

`\line{A teď jsem roztažen po celém řádku.}`

vytvoří čtyři řádky

Teď jsem vlevo nahoře.

Teď jsem uprostřed.

Teď jsem vpravo.

A teď jsem roztažen po celém řádku.

Jiný typ umístování mezer obdržíme pomocí řídicího slova `\hfil`. To způsobí, že všechno zbývající volné místo na řádku se shromáždí na to místo, kam jsme `\hfil` umístili. Jestliže změním náš poslední příklad na `\line{A teď jsem roztažen \hfil po celém řádku.}`, obdržíme

A teď jsem roztažen po celém řádku.

Jestliže použijeme `\hfil` vícekrát, rozdělí se zbývající místo na odpovídající pozice stejným dílem. Tedy `\line{A teď \hfil jsem roztažen \hfil po celém řádku.}` dá ve vstupním souboru

A teď jsem roztažen po celém řádku.

▷ Cvičení 3.6 Vysázejte následující řádek:

levé křídlo levý útočník střední útočník pravý útočník pravé křídlo

▷ Cvičení 3.7 Vysázejte následující řádek

0 3/5 4/5 1

Pomocí řídicího slova `\hskip` můžeme provádět ve vodorovném směru obdobné úpravy jako ty, které se realizují užitím `\vskip` ve svislém směru.

▷ Cvičení 3.8 Co se stane, jestliže vstupní soubor bude obsahovat:

`\line{\hskip 1 in JEDNA \hfil DVA \hfil TŘI}`

Zarovnávání pravého okraje lze zrušit pomocí řídicího slova `\raggedright`. Později si ukážeme, jak to lze udělat jen v části sázeného textu.

### 3.4 Poznámky pod čarou

Obecný vzor pro vytváření poznámek pomocí  $\text{\TeX}$  je `\footnote{...}{...}`. Znak k označení poznámky se umísťuje do první dvojice závorek. Některé ze znaků jsou `\dag` (†), `\ddag` (‡), `\S` (§) a `\P` (¶). Text poznámky píšeme do druhé dvojice

závorek. Označení poznámek čísly je trochu složitější. Poznámka<sup>21</sup> umístěná na konci této strany pod číslem 21 byla vytvořena pomocí

```
\footnote{${}^{\{21\}}{Toto je poznámka umístěná na konci této
stránky jako poznámka 21.}
```

příčemž řídicí slovo `\footnote` a celý text poznámky byl umístěn bezprostředně za slovo „Poznámka“. Tato konstrukce je opravdu poněkud komplikovaná. Později uvidíme, proč se to tak dělá. K tomu však budeme muset vědět něco o sazbě matematiky. V tomto okamžiku jsme si jen ukázali, jak se to dělá.

TeXbook:  
117

▷ Cvičení 3.9 Vytvořte stránku textu s relativně dlouhou poznámkou pod čarou tak, aby obsahovala několik řádků.

▷ Cvičení 3.10 Vytvořte stránku textu se dvěma různými poznámkami.

### 3.5 V záhlaví a pod textem

Řádky pro název a čísla stránek, které se umísťují nad a pod hlavním textem, se vytvářejí pomocí řídicích slov `\headline={...}` a `\footline={...}`. Používají se stejně jako řídicí slovo `\line{...}` v obvyklém textu na stránce. Velmi užitečné řídicí slovo je `\pageno`, které reprezentuje aktuální číslo stránky. Tak například `\headline={\hfil \tenrm Strana \the\pageno}` způsobí, že se číslo stránky objeví v pravém horním rohu spolu se slovem „Strana“ (podívejte se na pravý horní roh této stránky).

TeXbook:  
252–253

Řídicím slovem `\pageno` můžete přiřadit vhodné hodnoty, pokud chcete vytvářený dokument číslovat jinou posloupností čísel nežli obvyklou: např. číslování pomocí římských číslic lze vytvářet užitím záporných čísel (`\pageno = -1` na začátku dokumentu způsobí číslování římskými číslicemi).

TeXbook:  
252

Různá záhlaví lze vytvořit pro sudé a liché stránky pomocí:

```
\headline={\ifodd \pageno {...}\else {...}\fi}
```

kde text v první složené závorce je určen pro stránky vpravo a ve druhé závorce umístěný text se bude tisknout na levé stránky.

▷ Cvičení 3.11 Vytvořte konstrukci pro číslování stránek dole ve středu stránky, přičemž číslo je po obou stranách obklopeno pomlčkami (em-dash).

---

<sup>21</sup> Toto je poznámka umístěná na konci této stránky jako poznámka 21.

### 3.6 Přeplněné a nenaplněné rámečky

Jedno z největších zklamání zažívá „čerstvý“ uživatel  $\TeX$ u při setkání s přeplněnými a nenaplněnými rámečky (box). I když je rámeček vcelku hezké české slovo, přimlouváme se za zavedení alternativního termínu „box“ i do češtiny — budeme ho v tomto textu dále běžně užívat. Je velmi únavné se hlášenými o přeplněných (overfull) a nenaplněných (underfull) boxech probírat ve vytvořeném souboru \*.log. Při interaktivním zpracování jsou hlášeny průběžně i na obrazovce. Přeplněné boxy jsou na výstupu značeny černými obdélníčky (anglicky slug, což je slimák nebo plž, vypadá takto:  $\blacksquare$ ) na pravém okraji textu. Pro slug se užívají zhusta různá obscurní jména. Je to proto, že se tyto „slimáci“ objevují dokonce i v případě, že je vstupní soubor v naprostém pořádku. Vysvětlíme si, proč se objevují a co lze s nimi dělat.

Existují boxy dvojího typu: hboxy a vboxy. Pověštině odpovídají uspořádání vodorovného textu do řádků a odstavců do formátů stránek. Připomínáme, že  $\TeX$  čte celý odstavec před rozhodováním, jak má být rozdělen na řádky (řádkový zlom). Je to lepší, než když se pracuje „po řádcích“ (tradiční způsob sázení z období „horké sazby“), neboť nepatrné vylepšení jednoho řádku v odstavci může mít za následek úplnou katastrofu při sazbě některého z následujících. Tato výhoda má ale i svůj rub: s každou opravou se může měnit vše v sazbě odstavce a někdy i více. Když uspořádáme slova do řádku, je nutné vložit na správná místa mezery pro zarovnání pravého okraje. Velmi rozměrné mezery mezi slovy jsou zřejmě nežádoucí; závadnost (badness) je míra ošklivosti vzhledu mezer, které jsme byli nuceni použít. Přesněji: „badness“ libovolného řádku je nějaké číslo mezi 0 (dokonalý řádek) a 10 000 (katastrofa!). Existuje parametr nazvaný `\hbadness`, jehož předdefinovaná hodnota je 1 000. Každý řádek, jehož chyba je větší než `\hbadness` je ohlášen jako „špatný“ (underfull nebo overfull hbox). Zvětšíme-li hodnotu parametru `\hbadness`, počet hlášených nenaplněných nebo přeplněných boxů se zmenší. Položíme-li `\hbadness = 10000`, nebudou nenaplněné či přeplněné hboxy vůbec hlášeny. Přeplněné boxy jsou kromě zápisu v chybovém hlášení vyznačovány i do textu slugem, protože i relativně větší přeplnění lze lehce přehlédnout. Pokročilejší uživatel zvládne opravu zpravidla jen na základě chybového hlášení (velikost přeplnění je hlášena), začátečník se zpravidla snaží zorientovat podle zmíněných optických značek v textu — již jste se rozhodli, jak jim budete říkat?

S přeplněním pracuje  $\TeX$  podobně: někdy je tolerantní a dovolí, aby byl řádek nepatrně delší, než je `\hsize`, a tak dosáhne lepšího vzhledu celého odstavce (řádky budou naplněny vyváženěji). Řídicí slovo `\tolerance` určuje, kdy tato situace nastane. Je-li řádek závadný, porovnává se přiřazená hodnota s hodnotou parametru `\tolerance` a  $\TeX$  přidá slovo do řádku i tehdy, překročí-li se `\hsize`. Jestliže je pak řádek trochu (ale ne o mnoho) delší, vysadí se bez ohlašování. Řídicí slovo `\hfuzz` určuje, jak velké překročení délky řádku je povoleno; předdefinovaná hodnota je `\hfuzz = 0.1 pt`. Každý řádek, který je jen o málo delší, než je `\hsize`, je zřejmě vážným problémem;  $\TeX$  umístí na pravý okraj řádku znamení  $\blacksquare$ , které je velmi výrazné. Opět se lze od hlášení přeplněných boxů osvobodit zvětšením hodnoty parametru `\tolerance`. S hodnotou `\tolerance = 10000` se přeplněné

boxy a ... ( — cenzurou zabaveno, čtenář si doplní sám podle vlastního uvážení a nálady) neobjeví. Předdefinovaná hodnota je `\tolerance = 200`.

Šířka znaku ■ je určena pomocí řídicího slova `\overfullrule`. Včleníte-li do vstupního souboru `\overfullrule = 0 pt`, nebudou znamení vůbec tištěna. Samozřejmě přeplněné boxy v textu eventuálně budou, ale nebudou tak nápadné.

Zvětšení parametru `\tolerance` můžeme ve verzi T<sub>E</sub>Xu 3.x kombinovat ještě s parametrem `\emergencystretch`. Kladná hodnota tohoto parametru (v jednotkách cm, mm, pt, ...) způsobí, že v případě, kdy T<sub>E</sub>X není schopen vysadit daný odstavec bez překročení dané tolerance, zkusí to ještě jednou s možností zvětšit mezery mezi slovy právě o hodnotu parametru `\emergencystretch`. Díky tomu se případná „špatnost“, ke které by určitě vedlo ruční odstranění přeplněného hboxu, rozloží do všech řádků odstavce.

Nyní již víme, proč jsou hlášeny nenaplněné a přeplněné boxy. Umíme i změnit četnost těchto hlášení změnou hodnot `\hbadness`, `\hfuzz` a `\tolerance`. Navíc je vcelku zřejmé, že malá hodnota `\hsize` pro šířku textu činí zlom řádků mnohem obtížnějším a zvětšuje počet hlášení. Jsou to ale hlášení informativní, která můžete ignorovat (na vlastní riziko).

Poněkud stranou byla ponechána otázka dělení slov — ve skutečnosti při hledání vhodného řádkového zlomu odstavce pracuje T<sub>E</sub>X ještě s dělicím algoritmem. Dodáme-li další možnosti pro dělení slova do souboru nebo dokonce do příslušné části programu, můžeme se zbavit hlášeného přeplnění. T<sub>E</sub>X totiž pracuje při lámání řádků s automatickým dělením slov v češtině nebo v angličtině — i pro řadu dalších jazyků existují dělicí algoritmy s relativně vysokou spolehlivostí — a to mu pomáhá nalézt vhodná místa pro dělení. Např. standardní (neobohacené) anglické dělení však nerozdělí slovo „database“. Je-li ve vstupním souboru `data\-base`, povoluje se toto slovo rozdělit na vyznačeném místě za druhým písmenem „a“. Vyskytuje-li se takové slovo ve vstupním souboru vícekrát, lze na začátek souboru napsat `\hyphenation{data-base}` a pak bude povoleno stejným způsobem dělit toto slovo v celém souboru. (I když je algoritmus pro *angličtinu* poměrně dokonalý, nedělí slova na všech možných místech; existují seznamy některých takových slov a v případě potřeby lze modifikovat i „dělicí vzorník“ tak, že se častěji užívané slovo se správným dělením do předpisu pro dělení přidá.) V souboru s hlášeními o zpracování souboru T<sub>E</sub>Xem jsou uváděny přeplněné řádky i s možnostmi dělení tak, jak je dává příslušný algoritmus; někdy je však nejlepším řešením pro odstranění nenaplněných a přeplněných boxů drobná úprava textu vstupního souboru. Pro češtinu existují dva soubory dělicích vzorů. Ten, který je šířen v C<sub>S</sub>T<sub>E</sub>Xu, pochází od Ladislava Lhotky. Jiný soubor, který vytvořil Petr Novák, byl přislíben k šíření v rámci C<sub>S</sub>TUGu. Zájemce o technické detaily dělení slov odkazujeme na Appendix H: Hyphenation Knuthovy knihy **The T<sub>E</sub>Xbook**).

Vcelku podrobně jsme se zabývali problematikou řádkového zlomu, tedy horizontální strukturou stránky. Existuje i analogická „vertikální problematika“. Nenaplnění (*underfull*) a přeplnění (*overfull*) hboxů indikuje, jak dobře je text uspořádán

v řádcích. Nenaplnění a přeplnění vboxů je hlášeno při stránkovém zlomu, tj. tehdy, když se z odstavců vytvářejí stránky. Tak např. veliká tabulka, kterou nelze v půli rozdělit, vytvoří hlášení o nenaplněném vboxu v okamžiku, kdy se dokončuje sazba stránky; toto hlášení je opět uloženo do souboru \*.log. Řídící slovo `\vbadness` se používá analogickým způsobem jako `\hbadness`, přičemž jeho hodnota ovlivňuje hlášení o stránkovém zlomu.

▷ Cvičení 3.12 Vezměte soubor s několika odstavci a zpracujte ho užitím různých (malých) hodnot `\hsize`. Sledujte hlášení o přeplněných boxech. Potom vše zopakujte s různými hodnotami `\hbadness`, `\hfuzz` a `\tolerance`.



## Kapitola 4

### {Skupiny, {skupiny, {a ještě skupiny}}}

---

Možnost členění textu do skupin (groups) umožňuje obrovské zjednodušení vstupních souborů. Každá nová skupina začíná symbolem { a je ukončena pomocí symbolem }. Změny, které uděláme uvnitř takové skupiny, ztrácejí platnost po opuštění skupiny. Tak např., jestliže ve vstupním souboru bude napsáno {\bf čtyři tučně tištěná slova}, levá složená závorka vymezuje začátek skupiny, řídicí slovo \bf provede změnu písma a pravá složená závorka ukončí platnost této změny. V tomto případě dostaneme **čtyři tučně tištěná slova**. Po ukončení práce se skupinou bude použito opět písma, kterého jsme užili naposled před vstupem do skupiny. Toto je (nejjednodušší) způsob vysázení několika slov odlišným písmem. Je dokonce možné skupiny zařazovat „do sebe“.

Jako další příklad si ukážeme dočasnou změnu rozměru textu. Např.

```
{
\hsize = 4 in
\parindent = 0 pt
\leftskip = 1 in
vytvoří odstavec, který je čtyři palce
:
(taková chyba se udělá velmi lehce).
\par
}
```

vytvoří odstavec, který je čtyři palce široký, s levým okrajem vzdáleným o další palec od levého okraje ke středu stránky. Tato změna je dočasná, pouze tento odstavec bude upraven tímto způsobem. Po ukončení práce se skupinou se  $\text{\TeX}$  vrátí k původnímu způsobu sazby. Je nutné si uvědomit, že jak \par, tak i prázdný řádek ukončí odstavec a je nutné jednu z těchto možností použít před ukončením skupiny. Kdybychom to neudělali,  $\text{\TeX}$  by se vrátil k původním parametrům užívaným bezprostředně před tímto odstavcem (taková chyba se udělá velmi lehce).

V minulé kapitole jsme se setkali s použitím řídicího slova \narrower, ale nevysvětlili jsme, jak se dá zúžit text jen jednoho nebo několika odstavců. Podle

předcházejícího vzoru to nyní jistě hravě zvládnete. Jestliže uijeme vhodné řídicí slovo (jako je např. `\centerline`), text mezi složenými závorkami za ním tvoří automaticky skupinu. Je-li ve vstupním souboru

```
\centerline{\bf Tučný titulek },
```

obdržíme tučně napsaný text

**Tučný titulek**

umístěný uprostřed řádku bez ohledu na to, jakého písma jsme užívali bezprostředně před použitím `\centerline`. Nemusíme přidávat další závorky, tj. psát

```
\centerline{{\bf Tučný titulek }}.
```

Prázdná skupina `{}` je velmi užitečná. Umožňuje mj. psaní akcentů bez vytištění akcentovaných písmen. Tak např. `\~{}` dá na výstupu vlnku (tildu) bez jakéhokoli písmene pod ní. Prázdné skupiny lze i využít k tomu, abychom zastavili  $\TeX$  v „požírání“ mezer: Tedy Používám  $\TeX$  často a velmi rád dá na výstupu mezeru za symbolem (logem) „ $\TeX$ “, zatímco i 10 obyčejných mezer na tomtéž místě logo  $\TeX$  od následujícího textu neoddělí.

$\TeX$ book:  
19–21

▷ Cvičení 4.1 Změňte rozměry právě jednoho odstavce na stránce použitím konstrukce se skupinami.

▷ Cvičení 4.2 V matematické angličtině se někdy používá slova „iff“ jako zkratky pro „if and only if“ (právě když, neboli slovensky „akk“). V tomto případě je lépe, když druhé písmeno „f“ *není* spojeno s předcházejícím ligaturou. Jak se to dá udělat? (Existuje několik řešení.)

Velmi jednoduše lze zapomenout jednu ze složených závorek, které vymezují skupinu. To může mít *nedozírné* následky. Jestliže se vám např. stane to, že na výstupu obdržíte dokument, ve kterém se náhle změní písmo na kurzívu, pravděpodobně jste zapomněli uzavřít některou skupinu. Máte-li nepárovou složenou závorku `{`,  $\TeX$  vás na to upozorní v chybovém hlášení: (`\end occurred inside a group at level 1`). Nepárová pravá složená závorka `}` je ohlášena jako `! Too many }'s`.

Jak se dají sledovat závorky u složitých kombinací skupin? Pro větší přehlednost zkuste umístit levou složenou závorku na samostatný řádek a totéž udělejte s odpovídající pravou složenou závorkou. Je-li závorek víc a jsou do sebe vloženy, udělejte s vnitřními totéž, ale posuňte je obě o několik mezer doprava. Posuňte o stejný počet mezer vpravo i text mezi závorkami — nebude to mít na výsledný text vliv,  $\TeX$  mezery na začátku řádku bude ignorovat. Pozor, někdy je třeba „ohlídat“ mezery za závorkami. V takto upraveném vstupním textu jsou odpovídající složené závorky umístěny velmi přehledně a snadno se nalézají a kontrolují. S dobrým editorem vložíte nejprve najednou dvojici řádek s oběma odpovídající-

cími závorkami a pak se stejným automaticky řízeným posuvem vpravo vkládáte příslušný text.

▷ Cvičení 4.3 K napsání věty “Začneme s antikvou, *teď přejdeme na kurzívu* a nyní opět k antikvě” jsme v Kapitole 2 použili pro vstup “‘Začneme s antikvou, \it teď přejdeme na kurzívu\ / \rm a nyní opět k antikvě’”. Použijte nyní konstrukce s využitím skupin.

K nesporným výhodám T<sub>E</sub>Xu patří i možnost členit vstupní T<sub>E</sub>Xt (to je schválnost, ale v zahraničí je tento druh humoru mezi přívrženci T<sub>E</sub>Xu oblíben) tak, že je velmi přehledný. Přitom toto členění neovlivní výslednou sazbu. Příklad: text z ukázky o používání řídicího slova \item lze upravit ve vstupním souboru takto

```
\parskip = 0 pt
\parindent = 30 pt
\noindent
```

Vyřešte následující úkoly:

```
\item {(1)} Připravte se a~maximálně se soustředte.
      Cesta k~řešení úkolů je někdy namáhavá a~svízelná.
      Najděte nejprve odpověď na otázku (2)!
\item {(2)} Najděte odpověď na otázku (3)!
\item {(3)} Najděte odpověď na otázku (3a)!
      \itemitem {(3a)} Nejste-li unaveni, pokračujte až do
      úplného vyřešení všech úkolů.
      Nyní najděte odpověď na otázku (3b)!
      \itemitem {(3b)} Najděte odpověď na otázku (1)!
```

Přitom opět dostanete

Vyřešte následující úkoly:

- (1) Připravte se a maximálně se soustředte. Cesta k řešení úkolů je někdy namáhavá a svízelná. Najděte nejprve odpověď na otázku (2)!
- (2) Najděte odpověď na otázku (3)!
- (3) Najděte odpověď na otázku (3a)!
  - (3a) Nejste-li unaveni, pokračujte až do úplného vyřešení všech úkolů. Nyní najděte odpověď na otázku (3b)!
  - (3b) Najděte odpověď na otázku (1)!

## Kapitola 5

### Žádný strach z matematiky!

---

Když se sází matematika, je  $\text{T}_{\text{E}}\text{X}$  ve svém živlu. Pravidla pro její sázení jsou složitá, ale schopnost  $\text{T}_{\text{E}}\text{X}$ u je brát všechny v úvahu dává možnost vytvářet krásné a vysoce kvalitní matematické dokumenty. Hodláte-li psát texty obsahující matematické symboly, naleznete v této části veškeré základní informace o vytváření atraktivního matematického výstupu téměř jakéhokoli typu.  $\text{T}_{\text{E}}\text{X}$  lze samozřejmě užívat i bez matematiky — pokud je to váš případ — postačí vám pravděpodobně seznámit se jen se dvěma následujícími částmi kapitoly.

#### 5.1 Mnoho nových symbolů

Matematický text se vkládá do normálního textu dvěma možnými způsoby: může být začleněn *do řádků*, tj. tvořit část obyčejného textu, nebo jsou formule umísťovány na samostatné řádky vždy doprostřed (centrovány). V obou případech se výsledek pro tutéž formuli může lišit ve velikosti mezer mezi symboly, a případně i v umístění. V řádku umístěná rovnost  $\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6}$  se zřejmě liší od stejné rovnosti, pokud je na samostatném řádku:

$$\sum_{k=1}^{\infty} \frac{1}{k^2} = \frac{\pi^2}{6}.$$

Poznamenáváme, že u nás platná pravidla matematické sazby zakazují použití vodorovných zlomkových čar v textových řádcích a doporučují sazbu na samostatný řádek nebo náhradu zlomkových čar šikmými lomítky s použitím záporných exponentů. Proto by v textu měla formule z předcházejícího příkladu mít tvar  $\sum_{k=1}^{\infty} (1/k^2) = \pi^2/6$  nebo tvar  $\sum_{k=1}^{\infty} k^{-2} = \pi^2/6$ . Výjimku tvoří sazba číselných zlomků, nejlépe je však zjistit např. mínění redaktora odpovědného za sazbu textu apod.

Protože jsou velikosti mezer i symboly pro psaní matematiky zcela odlišné od normálního textu, musíme  $\text{T}_{\text{E}}\text{X}$  informovat o tom, kdy začínáme sázet matematiku a kdy jde o normální text. To se dělá pomocí symbolu  $\$$ . Přesněji, do textu (ne na samostatný řádek) se matematika sází mezi dva znaky pro US dolar ( $\$. \dots \$$ ) a to, co má být na samostatném řádku, se umísťuje mezi „dvojitě dolary“:  $\$\$ \dots \$\$$  (sazba matematiky je drahá!). Proto tedy  $\$x = y+1\$$  dává  $x = y + 1$  v řádku, zatímco  $\$\$x = y+1\$\$$  vytvoří

$$x = y + 1$$

na samostatném řádku (display math mode).

Velikost mezer je v obou případech zcela řízena  $\TeX$ em. Přidávání mezer do vstupního souboru ve vzorcích neovlivní nijak výsledek na výstupu. Když budete potřebovat vytvořit si trochu místa nebo umístit nějaký text přímo do matematické formulky, lze to udělat jeho umístěním do hboxu (zatím nemusíte vědět, proč se to tak dělá): `\hbox{...}`. Je to však velmi užitečné zejména při sázení vzorců na samostatný řádek.

Nebývá to sice zpravidla potřeba nebo alespoň ne příliš často, přece jen je ale vhodné seznámit se s některými řídicími slovy, která se při vkládání mezer používají. Jsou podle velikosti seřazena v následující tabulce:

### Přidávání mezer do matematického textu

Jméno	Řídicí posloupnost	←Velikost→
Dvojitý (double) quad	<code>\qqquad</code>	
Quad	<code>\quad</code>	
Mezera (space)	<code>\quad</code>	
Velká (thick) mezera	<code>\; </code>	
Střední (medium) mezera	<code>\&gt; </code>	
Malá (thin) mezera	<code>\, </code>	
Záporná malá mezera	<code>\! </code>	

Když si prohlédnete, jak vypadá záporná malá mezera (negative thin space), všimnete si, že se obě vymezující značky překrývají. Ostatní řídicí sekvence do sazby místo přidávají, kdežto záporná malá mezera ho ubírá. Dokonce lze dosáhnout i toho, že se symboly budou překrývat.

▷ Cvičení 5.1 Vysázejte tento vzoreček:  $C(n, r) = n!/r!(n - r)!$ . Všimněte si pečlivě, jak vypadají mezery ve jmenovateli.

Mezi dolary by se neměl vyskytnout žádný prázdný řádek, protože  $\TeX$  předpokládá, že celý matematický text je v jednom odstavci. Když to nevědomky uděláte, objeví se chybové hlášení. Vzhledem k okolnosti, že zapomenout nějaký ten dolar (přesněji: zapomenout ukončit matematický režim) je velmi lehké a jistě se vám to stane, je tato úprava docela rozumná — uchrání vás od toho, že by zbytek vašeho textu byl vysázen tak, jako se sází matematika.

Většina matematického textu se vkládá stejně pro umístění do textu i pro umístění na zvláštní řádek (samozřejmě až na počet znaků pro dolar na začátku a na konci). Výjimky si vysvětlíme později (číslování rovnic vpravo nebo vlevo, vertikální zarovnávání více rovnic podle rovnítka aj.).

Při sázení matematiky se vyskytuje mnoho dalších nových symbolů. Většina z nich se nalézá na klávesnici počítače a lze jich použít přímo. Jsou to symboly

+ - / \* = ' | < > ( ); svislá čára na klávesnici bývá přerušena a začátečníci si ji pletou s dvojtečkou. Po vytištění v matematickém textu vypadají tyto symboly takto: + - /\* = ' | < > ( ).

▷ Cvičení 5.2 Vysázejte rovnici  $a + b = c - d = xy = w/z$  do textu a pak ještě jednou na samostatný řádek.

▷ Cvičení 5.3 Udělejte totéž s rovnicí  $f(x, y) = x' + x(x + y)$ .

Mnoho dalších symbolů se podle očekávání vkládá pomocí řídicích slov. Jsou k dispozici všechna řecká písmena — přehled těch nejčastěji používaných podává následující tabulka (některá velká řecká písmena jsou shodná s latinkovými, např. A, B aj.).

TeXbook:  
434

### Řecká písmena

$\alpha$	<code>\alpha</code>	$\beta$	<code>\beta</code>	$\gamma$	<code>\gamma</code>	$\delta$	<code>\delta</code>
$\epsilon$	<code>\epsilon</code>	$\varepsilon$	<code>\varepsilon</code>	$\zeta$	<code>\zeta</code>	$\eta$	<code>\eta</code>
$\theta$	<code>\theta</code>	$\vartheta$	<code>\vartheta</code>	$\iota$	<code>\iota</code>	$\kappa$	<code>\kappa</code>
$\lambda$	<code>\lambda</code>	$\mu$	<code>\mu</code>	$\nu$	<code>\nu</code>	$\xi$	<code>\xi</code>
$o$	<code>o</code>	$\pi$	<code>\pi</code>	$\rho$	<code>\rho</code>	$\varrho$	<code>\varrho</code>
$\sigma$	<code>\sigma</code>	$\varsigma$	<code>\varsigma</code>	$\tau$	<code>\tau</code>	$\upsilon$	<code>\upsilon</code>
$\phi$	<code>\phi</code>	$\varphi$	<code>\varphi</code>	$\chi$	<code>\chi</code>	$\psi$	<code>\psi</code>
$\omega$	<code>\omega</code>	$\Gamma$	<code>\Gamma</code>	$\Delta$	<code>\Delta</code>	$\Theta$	<code>\Theta</code>
$\Lambda$	<code>\Lambda</code>	$\Xi$	<code>\Xi</code>	$\Pi$	<code>\Pi</code>	$\Sigma$	<code>\Sigma</code>
$\Upsilon$	<code>\Upsilon</code>	$\Phi$	<code>\Phi</code>	$\Psi$	<code>\Psi</code>	$\Omega$	<code>\Omega</code>

▷ Cvičení 5.4 Vysázejte v textu  $\alpha\beta = \gamma + \delta$  a pak totéž na samostatný řádek.

▷ Cvičení 5.5 Vysázejte v textu  $\Gamma(n) = (n - 1)!$  a pak totéž na samostatný řádek.

Někdy je zapotřebí umístit nad a pod symboly další znaky. V matematickém módu se pro to užívají jiná řídicí slova než v nematematické části textu. Řídicí slova z normálního textu nelze užívat pro sázení matematiky a obráceně. Následující tabulku nepřekládáme, nemělo by to rozumný smysl (používáme poněkud nepřesného označení slovem akcent kvůli analogii s anglickým „math accent“):

TeXbook:  
135–136

### Matematické akcenty

$\hat{o}$	<code>\hat o</code>	$\check{o}$	<code>\check o</code>	$\tilde{o}$	<code>\tilde o</code>
$\acute{o}$	<code>\acute o</code>	$\grave{o}$	<code>\grave o</code>	$\dot{o}$	<code>\dot o</code>
$\ddot{o}$	<code>\ddot o</code>	$\breve{o}$	<code>\breve o</code>	$\bar{o}$	<code>\bar o</code>
$\vec{o}$	<code>\vec o</code>	$\widehat{abc}$	<code>\widehat {abc}</code>	$\widetilde{abc}$	<code>\widetilde {abc}</code>

Symbole pro binární operace kombinují dva matematické objekty a vytvářejí nový matematický objekt. Obyčejné sčítání a násobení např. kombinují dvě čísla a výsledkem je zase číslo — jsou to příklady binárních operací. Sázíme-li binární operátory jako  $+$  a  $\times$ , TeX automaticky upravuje správné mezerování po jejich stranách. Následující tabulka dává přehled binárních operátorů:

### Binární operátory

$\cdot$	<code>\cdot</code>	$\times$	<code>\times</code>	$*$	<code>\ast</code>	$\star$	<code>\star</code>
$\circ$	<code>\circ</code>	$\bullet$	<code>\bullet</code>	$\div$	<code>\div</code>	$\diamond$	<code>\diamond</code>
$\cap$	<code>\cap</code>	$\cup$	<code>\cup</code>	$\vee$	<code>\vee</code>	$\wedge$	<code>\wedge</code>
$\oplus$	<code>\oplus</code>	$\ominus$	<code>\ominus</code>	$\otimes$	<code>\otimes</code>	$\odot$	<code>\odot</code>

Relace vypovídá něco o vlastnosti (dvou) matematických objektů. Již víme, jak se vyjádří, že jsou si dva objekty rovny a jak se to vysází. Stejně umíme vysázet i srovnání pomocí znamének pro „je větší“ a „je menší“, protože příslušné znaky jsou téměř na všech klávesnicích. K negování relace se používá řídicí slovo `\not`, které se klade před označení relace. V následující tabulce jsou některé relace:

### Relace

$\leq$	<code>\leq</code>	$\not\leq$	<code>\not \leq</code>	$\geq$	<code>\geq</code>	$\not\geq$	<code>\not \geq</code>
$\equiv$	<code>\equiv</code>	$\not\equiv$	<code>\not \equiv</code>	$\sim$	<code>\sim</code>	$\not\sim$	<code>\not \sim</code>
$\simeq$	<code>\simeq</code>	$\not\simeq$	<code>\not \simeq</code>	$\approx$	<code>\approx</code>	$\not\approx$	<code>\not \approx</code>
$\subset$	<code>\subset</code>	$\subseteq$	<code>\subseteq</code>	$\supset$	<code>\supset</code>	$\supseteq$	<code>\supseteq</code>
$\in$	<code>\in</code>	$\ni$	<code>\ni</code>	$\parallel$	<code>\parallel</code>	$\perp$	<code>\perp</code>

▷ Cvičení 5.6 Vysázejte:  $\vec{x} \cdot \vec{y} = \langle \vec{x}, \vec{y} \rangle = 0$ , právě když  $\vec{x} \perp \vec{y}$ .

▷ Cvičení 5.7 Vysázejte:  $\vec{x} \cdot \vec{y} = \langle \vec{x}, \vec{y} \rangle \neq 0$ , právě když  $\vec{x} \not\perp \vec{y}$ .

V další tabulce jsou uvedeny některé matematické symboly:

### Různé symboly

$\aleph$	<code>\aleph</code>	$\ell$	<code>\ell</code>	$\Re$	<code>\Re</code>	$\Im$	<code>\Im</code>
$\partial$	<code>\partial</code>	$\infty$	<code>\infty</code>	$\parallel$	<code>\parallel</code>	$\angle$	<code>\angle</code>
$\nabla$	<code>\nabla</code>	$\backslash$	<code>\backslash</code>	$\forall$	<code>\forall</code>	$\exists$	<code>\exists</code>
$\neg$	<code>\neg</code>	$\flat$	<code>\flat</code>	$\natural$	<code>\natural</code>	$\sharp$	<code>\sharp</code>

▷ Cvičení 5.8 Vysázejte:  $(\forall x \in \mathbb{R})(\exists y \in \mathbb{R}) y > x$ .

## 5.2 Zlomky

Číselný zlomek lze vysázet dvojím způsobem: lze to udělat buď ve tvaru  $1/2$  nebo ve tvaru  $\frac{1}{2}$ . V prvním případě nepoužíváme žádnou řídicí posloupnost a napíšeme jednoduše  $1/2$ . Ve druhém případě se použije řídicí slovo `\over` podle vzoru: `\langle čitatel \rangle \over \langle jmenovatel \rangle`. Tedy  $\frac{a+b}{c+d}$  dává

$$\frac{a+b}{c+d}$$

TeXbook:  
139–140

▷ Cvičení 5.9 Vysázejte:  $\frac{a+b}{c} \quad \frac{a}{b+c} \quad \frac{1}{a+b+c} \neq \frac{1}{a} + \frac{1}{b} + \frac{1}{c}$ .

▷ Cvičení 5.10 Vysázejte: Najděte body, ve kterých platí

$$\frac{\partial}{\partial x} f(x, y) = \frac{\partial}{\partial y} f(x, y) = 0$$

## 5.3 Indexy horní a dolní

Vysázet v TeXu dolní a horní indexy je velmi jednoduché. K vyznačení toho, že následující znak je dolním, resp. horním indexem, se používá řídicích znaků `_`, resp. `^`. Tak např.  $x^2$  dává  $x^2$  a  $x_2$  dává  $x_2$ . Má-li být dolním nebo horním indexem více znaků, vložíme je do složených závorek. Abychom tedy dostali  $x^{21}$ , resp.  $x_{21}$ , použijeme  $x^{\{21\}}$ , resp.  $x_{\{21\}}$ . Poznamenejme, že horní a dolní indexy jsou automaticky vytištěny v menší velikosti. Situace je však poněkud složitější pro horní a dolní indexy druhé úrovně. Nelze užít  $x_{2.3}$ , protože by to mohlo mít dvě interpretace:  $x_{\{2.3\}}$  a  $\{x_2\}_3$ ; dostáváme pak dva různé výsledky:  $x_{23}$  a  $x_{23}$ , z nichž pouze první odpovídá obvyklému použití dolních indexů v matematice. Musíme proto vícenásobné úrovně dolních a horních indexů popsat pomocí složených závorek. Horní a dolní indexy lze používat do libovolné úrovně.

TeXbook:  
128–130

Při použití horních a dolních indexů u téhož symbolu lze psát `_` a `^` v libovolném pořadí. Jak  $x_2^1$ , tak i  $x^{1.2}$  dávají  $x_2^1$ .

▷ Cvičení 5.11 Vysázejte:  $e^x \quad e^{-x} \quad e^{i\pi} + 1 = 0 \quad x_0 \quad x_0^2 \quad x_0^2 \quad 2^{x^x}$ .

▷ Cvičení 5.12 Vysázejte:  $\nabla^2 f(x, y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$ .

Řady a integrály se sázejí podobným způsobem. Použijeme-li  $\sum_{k=1}^n k^2$ , dostaneme  $\sum_{k=1}^n k^2$ ; použití  $\int_0^x f(t) dt$  dává  $\int_0^x f(t) dt$ .

TeXbook:  
144–145



Tento způsob sazby se objevuje i ve výrazech obsahujících limity. Po vložení  $\lim_{x \rightarrow 0} x^x = 1$  do vstupního souboru dostanete  $\lim_{x \rightarrow 0} x^x = 1$ .

▷ Cvičení 5.13 Vysázejte následující výraz:  $\lim_{x \rightarrow 0} (1+x)^{\frac{1}{x}} = e$ .

▷ Cvičení 5.14 Vysázejte následující text: Mohutnost množiny  $(-\infty, \infty)$  je  $\aleph_1$ .

Je vhodné uvést návod, jak zajistit, aby integrály vypadaly ještě trochu lépe. Podívejte se na rozdíl mezi  $\int_0^x f(t) dt$  a  $\int_0^x f(t) dt$ . Ve druhém případě následuje po  $f(t)$  malá dodatečná mezera a vzhled po vytištění je lepší; k vytvoření této dodatečné mezery jsme použili řídicí symbol `\,`.

▷ Cvičení 5.15 Vysázejte následující integrál:  $\int_0^1 3x^2 dx = 1$ .

V české matematické sazbě je zvykem (a patrně i normou), že písmena  $d$  (diferenciál),  $e$  (Eulerovo číslo) a  $i$  (imaginární jednotka) se (vcelku logicky) sázejí ve vzorcích antikvou, protože kurzíva je vyhrazena pro matematické proměnné. Anglosasové si s tím tolik hlavu nelámou, protože z kontextu se většinou velice snadno pozná, jde-li o proměnnou či nikoli.

## 5.4 Odmocniny, druhé mocniny a vůbec

K vysázení druhé odmocniny stačí použít konstrukci

`\sqrt{...}`,

takže `\sqrt{x^2+y^2}` ve vstupním souboru dává  $\sqrt{x^2 + y^2}$ . Všimněte si, že `\TeX` se stará nejen o umístění symbolů, ale i o výšku a délku znamení odmocniny. K vysázení třetích nebo jiných odmocnin se používají řídicí slova `\root` a `\of`. Použitím `\root n \of {1+x^n}` dostanete  $\sqrt[n]{1+x^n}$ .

`\TeXbook:`  
130–131

Jinou možností je užití řídicího slova `\surd`; pomocí `\surd 2` dostaneme na výstupu  $\sqrt{2}$ .

▷ Cvičení 5.16 Vysázejte:  $\sqrt{2} \quad \sqrt{\frac{x+y}{x-y}} \quad \sqrt[3]{10} \quad e^{\sqrt{x}}$ .

▷ Cvičení 5.17 Vysázejte:  $\|x\| = \sqrt{x \cdot x}$ .

## 5.5 Linky — nad i pod

K umístění vodorovné linky nad nebo pod matematickými výrazy (tedy „nadtržení“ a podtržení) se používá konstrukcí `\overline{...}` a `\underline{...}`.

Užijeme-li tedy konstrukce  $\overline{x+y}=\overline{x} + \overline{y}$ , dostaneme  $\overline{x+y} = \overline{x} + \overline{y}$ . Upozorňujeme však na to, že linky nad písmeny jsou v různých výškách, a proto je opatrnost namístě. Konstrukce  $\overline{\strut x}$  výšku linky nad  $x$  trochu zvětší.

TeXbook:  
130–131

K podtržení nematematického textu se používá  $\underbar{\dots}$ .

▷ Cvičení 5.18 Vysázejte:  $\underline{x}$   $\overline{y}$   $\overline{x+y}$ .

## 5.6 Oddělovače velké a malé

Nejužívanější matematické oddělovače jsou závorky (kulaté, hranaté a složené). Jak jsme již viděli, lze je vysázet např. pomocí  $( ) [ ] \{ \}$  (což dává  $() [] \{\}$ ). Větší oddělovače někdy zpřehlední matematické výrazy jako např. v následující formuli

$$(a \times (b + c))((a \times b) + c).$$

Ke zvětšení levých oddělovačů se před nimi použijí řídicí slova  $\bigl$ ,  $\Bigl$ ,  $\biggl$  nebo  $\Biggl$ ; podobně  $\bigr$ ,  $\Bigr$ ,  $\biggr$  a  $\Biggr$  se používají pro zvětšení pravých oddělovačů. Užitím  $\$ \Bigl [ \$ a \$ \Bigr ] \$$  tedy dostaneme  $\left[ a \right]$ .

TeXbook:  
145–147

Následující tabulka umožňuje porovnat velikost některých oddělovačů.

### Oddělovače různých velikostí

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

Chcete-li, můžete TeX nechat vybrat velikost oddělovačů automaticky umístěním řídicích slov  $\left$  a  $\right$  před příslušné oddělovače. Použití konstrukce  $\left[ \dots \right]$  tedy způsobí, že text bude vložen do hranatých závorek vhodné velikosti. **Upozornění:** každému oddělovači s  $\left$  bezpodmínečně **musí** odpovídat oddělovač s  $\right$  (a to i uvnitř každé grupy). Konstrukce pro absolutní hodnotu zlomku vypadá např. takto:

TeXbook:  
148

$\$ \$ \left| \{ a+b \over c+d \} \right| . \$ \$$

Na výstupu dává

$$\left| \frac{a+b}{c+d} \right|.$$

Přehled užívaných oddělovačů podává tabulka:

### Matematické oddělovače

(	(	)	)	[	[
]	)	{	\{	}	\}
\lfloor	\rfloor	\lceil	\rceil	\lceil	\rceil
\lceil	\rceil	\langle	\rangle	\langle	\rangle
/	/	\	\backslash		
\	\	\uparrow	\Uparrow	\uparrow	\Uparrow
\downarrow	\Downarrow	\downarrow	\Downarrow	\downarrow	\Downarrow
\Updownarrow	\Updownarrow				

▷ Cvičení 5.19 Vysázejte  $\lfloor x \rfloor - 1 < x$ .

## 5.7 Ach, ty speciální funkce

Existuje několik funkcí, jejichž označení se často vyskytuje v matematických textech. V rovnici typu  $\sin^2 x + \cos^2 x = 1$  jsou goniometrické funkce  $\sin$  a  $\cos$  vysázeny obyčejným písmem (na rozdíl od kurzívy používané v matematickém módu). To je obvyklá matematická konvence, jak vyznačit, že se jedná o funkci a nikoli o součin tří veličin. Uveďme tabulku řídicích slov pro psaní některých speciálních funkcí:

TeXbook:  
162

### Speciální matematické funkce

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

Pokud potřebujeme ve vzorcích jiné podobné výrazy, na které autor TeXu těžko mohl pamatovat, máme k dispozici konstrukci `\def\meno{\mathop{\rm meno}\nolimits}`. Na rozdíl od `\hbox{meno}` zaručuje předchozí konstrukce vhodné mezery podle místa použití. Vynecháme-li příkaz `\nolimits`, budou případné indexy či exponenty ve vzorcích na samostatném řádku vysazeny pod, resp. nad

jménem takto definovaného „operátoru“. Při psaní české matematiky můžeme takto definovat  $\operatorname{tg}$ ,  $\operatorname{cotg}$ ,  $\operatorname{arctg}$  a další matematické funkce, jejichž jména se v češtině liší od anglických názvů.

▷ Cvičení 5.20 Vysázejte:  $\sin(2\theta) = 2 \sin \theta \cos \theta$   $\cos(2\theta) = 2 \cos^2 \theta - 1$ .

▷ Cvičení 5.21 Vysázejte:

$$\int \csc^2 x \, dx = -\cot x + C \quad \lim_{\alpha \rightarrow 0} \frac{\sin \alpha}{\alpha} = 1 \quad \lim_{\alpha \rightarrow \infty} \frac{\sin \alpha}{\alpha} = 0.$$

▷ Cvičení 5.22 Vysázejte:

$$\tan(2\theta) = \frac{2 \tan \theta}{1 - \tan^2 \theta}.$$

## 5.8 Na vědomost se dává!

Existuje speciální makro, které se užívá téměř v každém matematickém článku a které je natolik zvláštní, že vyžaduje i zvláštní vysvětlení. Jedná se o makro `\proclaim`. Používá se ho při sazbě vět, důsledků, tvrzení a podobně. Odstavec následující po `\proclaim` se rozdělí na dvě části: první částí je text od začátku odstavce až po první tečku včetně a druhou část tvoří zbytek. Základní myšlenkou je, aby první část byla něco jako „Věta 1.“ nebo „Důsledek B.“ Tvrzení věty nebo důsledku tvoří druhou část. Ukažme si to na příkladu:

TeXbook:  
202-203

`\proclaim` Věta 1. Mezi slepými je jednooký králem.

dává

**Věta 1.** *Mezi slepými je jednooký králem.*

Tvrzení věty může samozřejmě obsahovat i matematické výrazy (také může být složeno z více gramatických vět).

▷ Cvičení 5.23 Vysázejte:

**Tvrzení 1.**  $\sum_{i < j}^n |X_i - X_j| = 0$ , právě když  $X_1 = \dots = X_n$ .

## 5.9 Matice

Matice se sázejí pomocí kombinací zarovnávacího symbolu `&` a řídicího slova `\cr`, sloužícího k vyznačení konce řádku. Konstrukce má tvar `$$\pmatrix{..}$$`. Mezi složené závorky se vypisují jednotlivé řádky matice, přičemž každý řádek je zakončen `\cr`. Prvky matice jsou odděleny pomocí `&`. Například: použijeme-li ve vstupním textu

TeXbook:  
176–178

```
$$\pmatrix{
a & b & c & d \cr
b & a & c+d & c-d \cr
0 & 0 & a+b & a-b \cr
0 & 0 & ab & cd \cr
}$$
```

dostaneme

$$\begin{pmatrix} a & b & c & d \\ b & a & c+d & c-d \\ 0 & 0 & a+b & a-b \\ 0 & 0 & ab & cd \end{pmatrix}$$

▷ Cvičení 5.24 Vysázejte

$$I_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Je možné též pracovat i s maticemi omezenými jinými oddělovači. Když použijeme `\matrix` místo `\pmatrix`, závorky budou vynechány, takže musíme použít explicitně oddělovače `\left` a `\right`. Ukažme si, jak lze změnit matici z posledního příkladu. Vložení do vstupního souboru (píšeme nyní vstup do řádku, což je nepřehledné, ale z hlediska úspory místa vhodné)

```
$$ \left |
\matrix{a & b & c & d \cr b & a & c+d & c-d \cr
0 & 0 & a+b & a-b \cr 0 & 0 & ab & cd \cr}
\right | $$
```

dostaneme na výstupu

$$\left| \begin{array}{cccc} a & b & c & d \\ b & a & c+d & c-d \\ 0 & 0 & a+b & a-b \\ 0 & 0 & ab & cd \end{array} \right|$$

Vynechání „otevírajícího“ a „uzavírajícího“ oddělovače je také možné; k tomu stačí použít `\left.` a `\right.` (upozorňujeme na použití tečky). Např. pro vytvoření formulky

$$F(x) = \begin{cases} \frac{\sin x}{x}, & x \neq 0 \\ 1, & x = 0 \end{cases}$$

použijeme konstrukce

```


$$F(x) = \left\{ \begin{array}{l} \frac{\sin x}{x}, & x \neq 0 \\ 1, & x = 0 \end{array} \right.$$


```

▷ Cvičení 5.25 Vysázejte

$$|x| = \begin{cases} x, & x \geq 0 \\ -x, & x \leq 0 \end{cases}$$

Toto cvičení lze také vysázet pomocí makra `\cases`.

TeXbook:  
175

Matice lze také sázet i v řádcích textu, vypadá to však značně ošklivě, pokud počet řádků matice není malý.

## 5.10 Samostatné rovnice

Všechny zatím uvažované matematické části textu vypadaly téměř stejně, ať jsme je vysázeli do řádku nebo samostatně (na zvláštní řádek). Nyní si všimneme situací, které jsou typické pouze pro samostatné rovnice (displayed equations).

První se týká zarovnávání víceřádkových rovnic nebo několika rovnic ve více řádcích. To se provádí pomocí zarovnávacího symbolu `&` a řídicích slov `\cr` a `\eqalign{...}`. Začneme řídicím slovem `\eqalign`, potom píšeme rovnice, které mají být zarovnané, a každou z nich ukončíme `\cr`. V každé rovnici musí být jeden zarovnávací symbol `&` k vyznačení místa, podle kterého se zarovná. Obvykle se zarovná podle znaků rovnosti, i když to není nutné. Například

TeXbook:  
190–192

```


$$\begin{aligned} a + b &= c + d \\ x &= w + y + z \\ m + n + o + p &= q \end{aligned}$$


```

dává po vytištění

$$\begin{aligned} a + b &= c + d \\ x &= w + y + z \\ m + n + o + p &= q \end{aligned}$$

Samostatné rovnice lze číslovat buď na pravém nebo levém okraji textu. Vyskytne-li se u samostatné rovnice řídicí slovo `\eqno`, potom se všechno, co po něm následuje, umístí na pravý okraj. Například `$$ x+y=z. \eqno (1)$$` dává

$$x + y = z. \tag{1}$$

K očíslování rovnice na levém okraji se místo `\eqno` použije `\leqno`.

Nakonec předpokládejme, že je nějaký text nutno umístit někam do samostatné rovnice (nebo vzorce). To lze udělat pomocí boxu typu `hbox`. Například `$$X=Y, \hbox{ právě když }x=y.$$` dává

$$X = Y, \text{ právě když } x = y.$$

▷ Cvičení 5.26 Vyřešte několik velmi zajímavých problémů, které naleznete na str. 180 a 181 knihy `TeXbook`.

## Kapitola 6

### Všechno v jednom řádku

---

Přání umístit doprostřed do nějakého textu tabulku není nic neobvyklého. Naštěstí to  $\text{T}_{\text{E}}\text{X}$  umí jednoduše zařídit. Existují dvě různé metody zarovnávání textu. První z nich používá tabulační prostředky a podobá se nastavení tabulačních zarážek na psacím stroji. Každý řádek se vyplňuje zvlášť podle nastavených tabulačních sloupců, ale s větší flexibilitou než na psacím stroji. Druhou metodou je metoda horizontálního zarovnávání, při které se celá tabulka vysází najednou pomocí předepsaného vzoru.

#### 6.1 Tab, tab, tabulka

K zarovnání údajů pomocí tabulačních prostředků je třeba nejprve nastavit tabulační pozice pomocí řídicího slova `\settabs`. Potom se řádek tabulky s těmito tabulačními zarážkami uvádí řídicím symbolem `\+` a ukončuje `\cr` (připomeňme, že skutečný počet mezer v řádku při jeho sazbě není rozhodující). Nejjednodušší způsob použití řídicího slova `\settabs` spočívá v umístění textu do stejných sloupců. Po `\settabs 4 \columns` se tabulační zarážky nastaví tak, že vzniknou čtyři stejné sloupce. Tabele se potom provádí pomocí zarovnávacího znaku `&`, po kterém dojde k posunutí k další tabulační zarážce. Tak například

$\text{T}_{\text{E}}\text{X}$ book:  
231

```
\settabs 3 \columns
\+ British Columbia & Alberta & Saskatchewan \cr
\+ Manitoba & Ontario & Quebec \cr
\+ New Brunswick & Nova Scotia & Prince Edward Island \cr
\+ && Newfoundland \cr
```

vytvoří tabulku kanadských provincií

British Columbia	Alberta	Saskatchewan
Manitoba	Ontario	Quebec
New Brunswick	Nova Scotia	Prince Edward Island
		Newfoundland

Poznamenejme, že některé tabulační zarážky je možné přeskočit a že není nutné v jednom řádku využívat všech tabulačních zarážek. K vytvoření stejné tabulky se čtyřmi sloupci stačí použít `\settabs 4 \columns` k novému nastavení tabulačních zarážek. Řádky z předchozího příkladu dají



British Columbia	Alberta	Saskatchewan
Manitoba	Ontario	Quebec
New Brunswick	Nova Scotia	Prince Edward Island
		Newfoundland

V tomto příkladu jsou sloupce samozřejmě užší. Při dalším experimentu s šesti sloupci dostaneme

British Columbia	Alberta	Saskatchewan
Manitoba	Ontario	Quebec
New Brunswick	Nova Scotia	Prince Edward Island
		Newfoundland

Prvky v tabulce se již překrývají. Je to proto, že  $\text{T}_{\text{E}}\text{X}$  přejde k následující tabulační zarážce, i když to znamená (na rozdíl od psacího stroje) vrátit se na řádku zpět.

Mezi seskupováním a tabelací je zajímavý vztah. Tabelační hodnoty nastavené pomocí `\settabs` platí, jak lze ostatně čekat, pouze v té skupině, ve které byly definovány. Nastavení tabulace lze tedy dočasně změnit seskupením do složených závorek. Navíc každá položka tabulky je vlastním seskupením. Můžeme tedy jednu položku vysázet tučně, např. pomocí `\bf` bez použití složených závorek. Dokonce je možné umístit položku ve sloupci doprostřed, vlevo nebo vpravo, anebo vyplnit sloupec linkou nebo tečkami. Každá položka má na svém konci implicitní `\hfil`, takže se ve sloupci vytiskne vlevo. Přidáme-li `\hfil` na začátek položky, potom se položka vytiskne uprostřed sloupce, stejně jako je tomu s řídicím slovem `\line`. Přidáme-li `\hfill` na začátek, položky se posunou vpravo (`\hfill` má podobnou funkci jako `\hfil`, absorbuje totiž nadměrné mezery; `\hfilll` lze také použít; řídicí slovo, které má více písmen `l`, má „přednost“). Ve vstupním souboru umístěný text

```
\settabs 4 \columns
+\hfil British Columbia & \hfill Alberta \qqquad &
  \hskip0.1in \bf Saskatchewan & Manitoba \cr
+\hfil Ontario & \hfill Quebec \qqquad &
  \hskip0.1in \bf New Brunswick & Nova Scotia \cr
+\hfil --- & \hfill * \qqquad &
  \hskip0.1in \bf Newfoundland & Prince Edward Island \cr
+\ \dotfill && \hrulefill & \cr
```

vytvoří tabulku, v níž je první sloupec vystředěn (centrován), druhý sloupec je zarovnán vpravo s následnou mezerou vytvořenou pomocí `\qqquad` a třetí sloupec je vytištěn tučně s posunutím `0.1 in` doprava. Čtvrtý sloupec je zarovnán vlevo. Řídicí slova `\dotfill` a `\hrulefill` vyplní dvě položky v posledním řádku. Dostáváme tedy

British Columbia	Alberta	<b>Saskatchewan</b>	Manitoba
Ontario	Quebec	<b>New Brunswick</b>	Nova Scotia
—	*	<b>Newfoundland</b>	Prince Edward Island

.....

Kdybychom volili umístění a sazbu rozmanitější pomocí

```
\settabs 4 \columns
\+\hfil British Columbia & \hfill Alberta \qqquad &
  \hskip 5mm \bf Saskatchewan & Manitoba \cr
\+\hfill Ontario & \hfil Quebec \hfil \qqquad &
  \hskip 3mm New Brunswick & Nova Scotia \cr
\+\ --- & \hfill * \qqquad &
  \hskip 1mm \bf Newfoundland & Prince Edward Island \cr
\+ \dotfill && \hrulefill & \cr
```

dostali bychom tabulku sice rozmanitou, avšak dosti ošklivou; přesto stojí za to si ji prohlédnout a uvědomit si význam jednotlivých položek

British Columbia	Alberta	<b>Saskatchewan</b>	Manitoba
Ontario	Quebec	New Brunswick	Nova Scotia
—	*	<b>Newfoundland</b>	Prince Edward Island

.....

▷ Cvičení 6.1 V tabulce kanadských provincií všechny položky ve sloupcích umístěte na střed sloupce.

Nastavení tabulačních zarážek lze provést mnohem rozmanitěji než jen pro stále stejně široké sloupce. Obecným vzorem je použití „vzorového řádku“ tvaru

```
\settabs \+ ... & ... & ... \cr.
```

Velikost zadaných mezer mezi zarovnávacími symboly & určuje polohu tabulačních zarážek. Například

```
\settabs \+ \hskip 1 in & \hskip 2 in & \hskip 1.5 in & \cr
```

nastaví první zarážku jeden palec od levého okraje, druhou zarážku ještě o dva palce vpravo a třetí zarážku o dalších 1,5 palce vpravo. Zkuste provést experiment s takto zadaným řídicím řádkem.

K vymezení vzdáleností mezi tabulačními zarážkami lze také použít text. Uvedme následující příklad vzorového řádku:

```
\settabs \+ \quad Provincie \quad & \quad Populace \quad
  & \quad Rozloha \quad & \cr
```

Sloupce v tabulce potom budou dostatečně široké, aby se do nich vešla záhlaví tabulky i s mezerami na obou stranách (zatím však text určuje pouze rozměry sloupců). Následuje trochu úplnější příklad:

```
\settabs
\+ Rok \quad & Cena \qquad & Dividendy & \cr
\+ \hfill Rok & \hfil Cena & \hfil Dividendy & \cr
\+ \hfill 1971 & \hfil 41--54 & \hfil \$2.60 & \cr
\+ \hfill 2 & \hfil 41--54 & \hfil \$2.70 & \cr
\+ \hfill 3 & \hfil 46--55 & \hfil \$2.87 & \cr
\+ \hfill 4 & \hfil 40--53 & \hfil \$3.24 & \cr
\+ \hfill 5 & \hfil 45--52 & \hfil \$3.40 & \cr
```

TeXbook:  
247

dává na výstupu

Rok	Cena	Dividendy
1971	41–54	\$2.60
2	41–54	\$2.70
3	46–55	\$2.87
4	40–53	\$3.24
5	45–52	\$3.40

Pár poznámek nakonec: tvorba tabulek pomocí `\settabs`, `\+` a `\cr` je velmi jednoduchá. Používá se v ní konstrukce s řídicím slovem `\halign`, se kterým se seznámíme blíže v následující části. Každý řádek tabulky je ve skutečnosti samostatnou tabulkou a nedáte-li celou tabulku do `\vbox{...}`, láme se při přechodu na další stránku po řádcích stejně jako text. Do tabulek nelze shora popsaným způsobem umisťovat poznámky pomocí `\footnote`. Zapomenete-li na `\cr`, dostanete velmi komplikované chybové hlášení; chyba bývá hlášena na některém z následujících řádků, což může ztížit její hledání.

▷ Cvičení 6.2 Tabulku z posledního příkladu umístěte více ke středu stránky.

▷ Cvičení 6.3 Jednou z možností, jak vystředit několik řádků, je použít řídicí sekvence `$$\vbox{...}$$`. Vystředíte tímto způsobem výše uvedenou tabulku. Je nutné vložit řádek `\settabs` do `\vbox` ?

▷ Cvičení 6.4 Vylepšete svůj poslední výsledek přidáním řádku pod záhlaví tabulky. Řídicí slovo `\hrule` doplní vodorovnou linku, pokud se uvede mezi dvěma řádky tabulky. Totéž zopakujte s řídicím slovem `\strut` po `\+` v řádku obsahujícím záhlaví sloupců; (`\strut` trochu zvětší mezery mezi řádky). Všimněte si dodatečné mezery, která vznikne.

TeXbook:  
82

▷ Cvičení 6.5 Vytvořte následující tabulku s dekadickým zarovnáním, t.j. s desetinnými čárkami pod sebou:

Švestky	8,00 Kč
Káva	12,00 Kč
Žvýkačka	10,00 Kč
Houby	16,50 Kč
Utopenec	12,00 Kč
Pomeranče	16,40 Kč
Slivovice	80,00 Kč
Rohlík	0,30 Kč
Cigarety	90,00 Kč
Uzený bůček	18,60 Kč
Pivo	<u>35,00 Kč</u>
Celkem	298,80 Kč

▷ Cvičení 6.6 Navrhněte vytvoření jednoduché tabulky typu obsahu knížky s použitím `\settabs` a s položkami, které vypadají přibližně takto:

Začínáme.....	1
Všechny velké a malé znaky.....	9

## 6.2 Vodorovné zarovnání s rafinovaným vzorem

Použití `\settabs` není nikterak složité a jakmile je vzor jednou nastaven, lze ho kdykoli použít v různých částech následujícího textu. Má však některé nedostatky. Především velikost sloupců musí být nastavena dříve, než jsou známy položky. A navíc, kdybychom chtěli vytisknout jeden sloupec tučně, museli bychom to vyznačit v každém řádku. Tyto potíže lze zmenšit použitím řídicího slova `\halign`.

T<sub>E</sub>Xbook:  
235–238

Srovnajte vstupní data a výslednou tabulku s podobnou tabulkou vytvořenou pomocí `\settabs` z předcházející části:

```
\halign{\hfil # & \hfill # \quad & \hskip3mm \bf # &
\hfil # \hfil \cr
British Columbia & Alberta & Saskatchewan & Manitoba \cr
Ontario & Quebec & New Brunswick & Nova Scotia \cr
--- & * & Newfoundland & Prince Edward Island\cr
\dotfill & & \hrulefill & \cr}
```

dává po zpracování T<sub>E</sub>Xem

British Columbia	Alberta	<b>Saskatchewan</b>	Manitoba
Ontario	Quebec	<b>New Brunswick</b>	Nova Scotia
—	*	<b>Newfoundland</b>	Prince Edward Island
.....		_____	

K sazbě hodně komplikovaných tabulek v  $\TeX$ u existují daleko silnější prostředky, zde se zabýváme jen základními postupy. Uvedme obecný vzor pro použití `\halign`:

```
\halign{ <vzorový řádek> \cr
<první samostatný řádek> \cr
<druhý samostatný řádek> \cr
:
<poslední samostatný řádek> \cr
}
```

Vzorový řádek i samostatné řádky jsou rozděleny na sekce (části) pomocí zarovnávacího symbolu `&`. Ve vzorovém řádku se v každé sekci používá řídicích slov stejným způsobem jako u řídicího slova `\line{}`. Tak například řídicí slovo `\hfil` lze použít k zarovnání vlevo, vpravo nebo k vystředění. Fonty lze změnit pomocí `\bf`, `\it` atp. Také text lze použít přímo ve vzorovém řádku. Dodejme, že v každé sekci se musí použít speciální symbol `#`, a to právě jednou. Každý samostatný řádek tabulky je potom nastaven tak, že každá jeho sekce se dosadí do příslušné sekce vzorového řádku v místě, kde se nachází řídicí symbol `#`. Použití ilustruje následující velmi jednoduchý příklad: zadáme-li ve vstupním souboru

```
\halign{\hfill #\quad & \hfill #\quad & \hfill #\quad\cr
Volský Žab      & Vožábinka & Radka Kvačka \cr
Velká Žbluňka  & Žblabuňka & Bill Drmla   \cr
Franta Kníkal  & Kvakule   & Brek         \cr}
```

dostaneme posléze na výstupu

```
Volský Žab   Vožábinka   Radka Kvačka
Velká Žbluňka  Žblabuňka   Bill Drmla
Franta Kníkal   Kvakule     Brek
```

Předvedme si trochu komplexnější a serióznější příklad; necht' je ve vstupním souboru zakódována tabulka následujícím způsobem:

```
\halign{\hskip 20mm $$$ & \hfil \quad # \hfil & \quad $$$
& \hfil \quad # \hfil \cr
\alpha & alfa & \beta & beta \cr
\gamma & gama & \delta & delta \cr
\epsilon & epsilon & \zeta & dzeta \cr
}
```

Ve vzorovém řádku se říká, že první sekce vytištěného textu bude dvacet milimetrů od levého okraje a zároveň v matematickém módu. Druhá sekce bude vystředěna

po přidání mezery zleva pomocí `\quad`. Podobně je tomu s třetí a čtvrtou sekcí. Výsledek potom vypadá takto:

$\alpha$	alfa	$\beta$	beta
$\gamma$	gama	$\delta$	delta
$\epsilon$	epsilon	$\zeta$	dzeta

V tomto případě se první samostatný řádek vytvoří tak, že se ve vzorovém řádku dosadí `\alpha` místo prvního `#`, `alfa` místo druhého `#`, `\beta` místo třetího `#` a `beta` místo čtvrtého `#`. Celý řádek se potom uschová pro konečné nastavení. Tímto způsobem se zpracují všechny řádky a teprve potom jsou nastaveny tak, aby byl každý sloupec dostatečně široký a vešly se do něj všechny jeho položky (důsledkem tohoto procesu je to, že tabulka s příliš mnoha položkami může způsobit, že  $\TeX$ u bude chybět paměť; je proto lépe dělat jen tabulky, které se vejdou přibližně na jednu stránku). Vzorový řádek tedy určuje vzor každé položky tabulky a samostatné řádky dosazují jednotlivé položky.

K oddělení položek tabulky se někdy používají vodorovné a svislé linky. K vlození vodorovných linek se používá `\hrule`, podobně jako v prostředí `\settabs`. Protože však nechceme zarovnat linku podle vytvořeného vzoru, použijeme řídicího slova `\noalign`. Vodorovné linky se tedy vkládají pomocí `\noalign{\hrule}`; svislé linky se vkládají pomocí `\vrule` buď ve vzorovém, nebo v samostatném řádku. Není to ale zcela jednoduché, musíme ještě něco dodat. Uvažujme náš poslední příklad a změníme vzorový řádek tak, aby vkládal svislé a vodorovné linky — provedeme to následujícím způsobem:

```
\halign{\hskip 20mm\vrule\quad $$$\quad & \vrule \hfil\quad # \hfil
        & \quad \vrule \quad $$$ \quad \vrule
        & \hfil \quad # \quad \hfil \vrule \cr
\noalign{\hrule}
\alpha & alfa & \beta & beta \cr
\noalign{\hrule}
\gamma & gama & \delta & delta \cr
\noalign{\hrule}
\epsilon & epsilon & \zeta & dzeta \cr
\noalign{\hrule}
}
```

Snadno nahlédneme, že konstrukce nedává zrovna právě to, co bychom chtěli:

	$\alpha$	alfa	$\beta$	beta
	$\gamma$	gama	$\delta$	delta
	$\epsilon$	epsilon	$\zeta$	dzeta

Má to celou řadu nedostatků: jsou to samozřejmě nejen protažené vodorovné linky, ale i text je příliš v boxech namačkan. Stejně jako v prostředí `\settabs` lze řádky udělat trochu vyšší vložením řídicího slova `\strut` do vzorového řádku.

Další potíží může nastat, když se sestavuje stránka, protože TeX může řádky od sebe trochu vzdálit, aby stránka vypadala lépe. To ale způsobí mezery ve svislých linkách. Bráníme se tomu tak, že v prostředí `\halign` použijeme řídicí slovo `\offinterlineskip`. Konečně se zbavíme linek trčících vlevo tím, že ve vzorovém řádku vynecháme `\hskip 2 in`. Výsledek tedy bude lepší, jestliže použijeme:

```
\moveright 20mm
\ vbox{\offinterlineskip
\ halign{\strut\vrule\quad $$$
\quad & \vrule \hfil\quad # \hfil
& \quad \vrule \quad $$$
\quad \vrule & \hfil\quad # \quad \hfil \vrule \cr
\ noalign{\hrule}
\alpha & alfa & \beta & beta \cr
\ noalign{\hrule}
\gamma & gama & \delta & delta \cr
\ noalign{\hrule}
\epsilon & epsilon & \zeta & dzeta \cr
\ noalign{\hrule}
}}
```

což dává

$\alpha$	alfa	$\beta$	beta
$\gamma$	gama	$\delta$	delta
$\epsilon$	epsilon	$\zeta$	dzeta

Chceme-li sestrojít tabulku s položkami v rámečcích, která bude na stránce vystředěna, provedeme to vložením `\vbox` do `\centerline{}`. Ukažme si jeden trik, který dává krásný výsledek. Vložíme-li `\vbox` mezi dvojité znaky pro dolar, tabulka bude vytištěna jako samostatný matematický text. Samozřejmě se nejedná o žádný matematický text, ale TeX přidá nad tabulkou i pod ní trochu místa a po vytištění to vypadá lépe. Vystředěnou tabulku s tímto příjemným přidáním mezer lze tedy vytvořit podle schématu, které lze popsat stručně v následujících čtyřech krocích:

- vlož `\vbox` mezi dva znaky dolaru;
- vlož `\offinterlineskip` a `\halign` do `\vbox`;
- do `\halign` vlož vzorový řádek se `\strut` na začátku a `\vrule` kolem každé položky;
- před každým řádkem tabulky i po něm by mělo být `\noalign{\hrule}`. A tedy je vzor:

```

$$
\ vbox{\ offinterlineskip
\ halign{
\ strut \ vrule # & \ vrule # & ... & \ vrule # \ vrule \ cr
\ noalign{\ hrule}
[ položka 1. sloupce] & [ položka 2. sloupce] & ...
& [ položka posledního sloupce] \ cr
\ noalign{\ hrule}
...
\ noalign{\ hrule}
[ položka 1. sloupce] & [ položka 2. sloupce] & ...
& [ položka posledního sloupce] \ cr
\ noalign{\ hrule}
}}
$$

```



## Kapitola 7

### Dál válejte sami

---

V této kapitole se naučíme vytvářet nová řídicí slova. Vytváření takových nových definic nazývaných makra je jedním z nejmocnějších prostředků, kterými  $\TeX$  disponuje. Jako první aplikaci si ukažme, jak nová definice nahrazením dlouhých řetězců krátkými může ušetřit spoustu psaní.

#### 7.1 Zkratka a dobře

K definování nových řídicích slov se používá řídicího slova `\def`. Nejjednodušeji se to dělá takto: `\def\jinejmeno{...}`. Kdykoli se potom řídicí slovo `\jinejmeno` objeví ve vašem zdrojovém souboru, bude nahrazeno tím, co je mezi složenými závorkami v definici. Řídicí slovo `\jinejmeno` musí samozřejmě vyhovovat konvenci o jménech řídicích slov, tj. musí to být řídicí slovo (samá písmena) nebo řídicí symbol (právě jedno „nepísmeno“). Předpokládejme například, že píšeme dokument, ve kterém se mnohokrát vyskytuje slovní spojení „Univerzita Karlova“. Pomocí `\def\uk{Univerzita Karlova }` definujeme novou řídicí sekvenci `\uk` a tu lze potom kdykoli použít. Věta, vložená pomocí `\uk je nejstarší univerzita ve střední Evropě.`, bude mít smysl. Jestliže takové řídicí slovo již existuje, nová definice ho nahradí (to se týká i řídicích slov předdefinovaných  $\TeX$ em, takže jistá opatrnost při volbě jmen je nezbytná). Každá definice má však jen lokální platnost (pouze ve skupině, ve které byla zavedena). Například

```
\def\uk{Univerzita Karlova }
\uk je nejstarší univerzita ve střední Evropě.
{\def\uk{Univerzita Komenského }
\uk je v Bratislavě. }
Proto je \uk starší než Univerzita Komenského.
```

dává

Univerzita Karlova je nejstarší univerzita ve střední Evropě. Univerzita Komenského je v Bratislavě. Proto je Univerzita Karlova starší než Univerzita Komenského.<sup>1</sup>

Jakmile je řídicí sekvence definována, může být použita v nových definicích. To je jedna z možností, jak vytvářet jednoduché vzory dopisů. Definujeme:

---

<sup>1</sup> Cvičení: prostudujte pečlivě vstupní text, jestli zjistíte, proč je za první i druhou větou dvojnásobná mezera. S `\nonfrenchspacing` to nemá nic společného!

```

\def\dopis{
\hrule
\medskip
\leftline{Vážený pan}
\leftline{\jmeno.}
\line{\hfil
V tomto krátkém dopise Vám oznamuji,
že se jmenujete \jmeno.\hfil}
\line {\hskip 2 in Se srdečným pozdravem,\hfil}
\line {\hskip 2 in Byrokrat v.r.\hfil}
\medskip
\hrule
}

```

Tento dopis používá řídicí slovo `\jmeno`, které zatím nebylo ještě definováno. Použijeme-li `\dopis`, v dopise se objeví stávající hodnota pro `\jmeno`. Proto

```

\def\jmeno{Jan Werich}
\dopis
\medskip
\def\jmeno{Jiří Suchý}
\dopis

```

vytvoří dvě kopie dopisu, každou se správným jménem:

---

Vážený pan  
Jan Werich.

V tomto krátkém dopise Vám oznamuji, že se jmenujete Jan Werich.  
Se srdečným pozdravem,  
Byrokrat v.r.

---



---

Vážený pan  
Jiří Suchý.

V tomto krátkém dopise Vám oznamuji, že se jmenujete Jiří Suchý.  
Se srdečným pozdravem,  
Byrokrat v.r.

---

Do složených závorek v `\def\jmeno{...}` bychom mohli vložit cokoli; třeba několik odstavců a mohli bychom použít i další řídicí sekvence (i když v tomto případě by to bylo poněkud podivné). Jako součást definice řídicího slova `\dopis` můžeme samozřejmě použít i `\vfill \eject`, což způsobí „vyjetí“ stránky.

▷ Cvičení 7.1 Navrhnete vzor dopisu, který používá řídicí slova `\jmeno`, `\ulice`, `\mesto`, `\okres` a `\psc`.

▷ Cvičení 7.2 Předpokládejme, že se chystáte upravit několik odstavců svého článku pomocí `\hangindent = 30 pt`, `\hangafter = 4` a `\filbreak` (nedělejte si zatím žádné starosti s tím, co tyto řídicí sekvence vlastně znamenají; v tuto chvíli je podstatné pouze to, že jakmile byly nastaveny, platí pouze v jednom odstavci). Definujte jedinou řídicí sekvenci `\nastavodst`, kterou lze vložit před každý odstavec, který má být upraven.

## 7.2 Vyplňování s parametry

Makra lze obecně používat i s parametry. Základní myšlenka je podobná jako u vzorového řádku v prostředí `\halign`. Podívejme se nejprve na případ s jedním parametrem. V tomto případě je řídicí sekvence definována tímto způsobem: `\def\noveslovo#1{...}`. Symbol `#1` se může vyskytnout (třeba i několikrát) ve složených závorkách v definici `\noveslovo`. To, co je ve složených závorkách, je vlastně vzor. Objeví-li se `\noveslovo{...}` v textu, potom se použije definice `\noveslovo` s novým obsahem ve složených závorkách vloženým do vzoru v místě každého výskytu `#1` v původní definici. Velmi důležité jsou mezery v původní definici; před otevřením složených závorek nepoužijte žádnou mezeru.

Jako příklad můžeme použít např. dopis z předcházející sekce upravený takto:

```
\def
\dopis#1{\hrule
\medskip
\leftline{Vážený pan}
\leftline{\bf#1.}
\line{\hfil
V tomto krátkém dopise Vám oznamuji,
že se jmenujete #1.\hfil }
\line{\hskip 2 in Se srdečným pozdravem,\hfil }
\line{\hskip 2 in Byrokrat v.r.\hfil }
\medskip
\hrule }
```

Nyní můžeme psát dopisy ještě efektivněji. Stačí napsat pouze

```
\dopis{Jan Werich }
\medskip
\dopis{Jiří Suchý }
```

a dostaneme

---

Vážený pan

**Jan Werich.**

V tomto krátkém dopise Vám oznamuji, že se jmenujete Jan Werich.  
Se srdečným pozdravem,  
Byrokrat v.r.

---

Vážený pan

**Jiří Suchý.**

V tomto krátkém dopise Vám oznamuji, že se jmenujete Jiří Suchý.  
Se srdečným pozdravem,  
Byrokrat v.r.

---

Nyní definujme nové makro `\def\tisktext#1{$$\vbox{#1}$$}` k tisku textu. Potom `\tisktext{...}` zařídí, že všechno mezi složenými závorkami se vloží do vlastního odstavce, odstavec se vystředí a přidá se vhodná mezera nahoře i dole, aby to po vytištění vypadalo pěkně. Tento odstavec (s `\hsize = 11 cm`) byl upraven právě pomocí makra `\tisktext`.

Parametr makra nemůže být delší než jeden odstavec. Jestliže se odstavec objeví jako část parametru, vznikne chyba. Na druhé straně náhodné vynechání uzavírací složené závorky má za následek, že celý zbytek souboru se vloží do parametru.

▷ Cvičení 7.3 Definujte makro `\uspesnost` tak, aby se po vložení `\uspesnost{89}` do zdrojového souboru vytiskla následující věta: Vaše úspěšnost je 89%.

Použití více parametrů není ve skutečnosti vůbec složitější. Obecný vzor pro definici nového řídicího slova se dvěma parametry je `\def\noveslovo#1#2{...}`. V definici mezi složenými závorkami se mohou vyskytnout `#1` i `#2`, třeba i několikrát. Objeví-li se v textu `\noveslovo{...}{...}`, potom se v definici nahradí `#1` tím, co je v první složené závorce, a `#2` tím, co je ve druhé složené závorce. Zde je příklad a jeho výsledek:

```
\def\hovor#1#2{#1 hovoří  
s #2.}  
\hovor{Jan}{Janou}  
\hovor{Jana}{Janem}  
\hovor{Jiří}{tebou}  
\hovor{Kdosi}{Janou}
```

dává

Jan hovoří s Janou. Jana hovoří s Janem. Jiří hovoří s tebou. Kdosi hovoří s Janou.

▷ Cvičení 7.4 Podobně jako v předcházejícím cvičení definujte makro `\uspesnost` tak, aby se po `\uspesnost{89}{85}` vytiskla následující věta: Tvoje úspěšnost v prvním testu byla 89% a ve druhém 85%.

Zdůrazněme, že před první složenou závorkou a mezi #1 a #2 v definici nebyla žádná žádná mezera. Uděláte-li ji, `TEX` bude interpretovat definici jinak, než bylo popsáno. Mezery nebo jiné znaky vložené mezi či za parametry #1 a #2 bude `TEX` chápat jako jakési oddělovače, které vymezují, kde jeden parametr končí a jiný začíná. To je ale složitější záležitost, kterou vám přenecháme k případným experimentům. Definice pro více než dva parametry je podobná. Řídící slovo se třemi parametry se definuje takto: `\def\noveslovo#1#2#3{...}`. Potom se může mezi složenými závorkami vyskytnout #1, #2 i #3. Vyskytne-li se v textu `\noveslovo{...}{...}{...}`, potom to, co je mezi dvojicí složených závorek, nahradí příslušný symbol v definici řídicího slova. Použitých parametrů může být nejvíce devět (#1 ... #9).

### 7.3 Kterýmkoli jiným jménem

Někdy je výhodné dát řídicímu slovu ještě jiné jméno. Např. místo řídicího slova `\centerline` bychom rádi používali `\stredlin`. To se dá zařídit pomocí řídicího slova `\let`. Po `\let \stredlin = \centerline` máme možnost používat nové (ale i původní) řídicí slovo. To je možné i u matematických jmen, např. `\tensor = \otimes`. Potom můžeme psát

`TEXbook:`  
206–207

`$$ (A \tensor B) (C \tensor D) = AC \tensor BD. $$`

Tak obdržíme

$$(A \otimes B)(C \otimes D) = AC \otimes BD.$$

▷ Cvičení 7.5 Definujte řídicí slova `\ll`, `\cl` a `\rl` tak, aby byla ekvivalentní s řídicími slovy `\leftline`, `\centerline` a `\rightline`.

Řídící slovo `\let` umožňuje uživateli nově pojmenovat řídicí sekvence a vytvořit si tak svou vlastní sadu řídicích sekvencí, kterou může používat místo původních řídicích slov, která jsou v `TEXu` předdefinována.

## Kapitola 8

### Chybovat je lidské

---

V jistém ohledu není T<sub>E</sub>X žádný zázrak. Na chybu ve vstupu reaguje chybovým hlášením na obrazovku (užíváte-li jej interaktivně) a také zápisem chyby do stejnojmenného souboru s extenzí `.log` (log file). Protože je T<sub>E</sub>X velmi složitý, skutečný výskyt nalezené a ohlášené chyby může být ukryt hluboko v programu; proto úplná chybová zpráva může být i velmi dlouhá a složitá. Nejen to, T<sub>E</sub>X se pokusí chyby napravit a podá o tom zprávu. Z tohoto důvodu může být čtení chybových hlášení pro začátečníka poněkud složité. Rozhodující je jen vědět, co je pro nás důležité a co lze klidně ignorovat. Podívejme se nyní na typické chyby a hlášení, která vytvářejí.

#### 8.1 Zapomenuté konce

První chyba, na kterou se podíváme, je chyba, kterou někdy udělá každý. Je to totiž vynechání `\bye` nebo `\end` na konci souboru. Používáte-li T<sub>E</sub>X interaktivně, na obrazovce se objeví

\*

a dál se nebude dít nic, protože jste neřekli, že se má skončit; T<sub>E</sub>X bude čekat na vstup z klávesnice.

Cokoli napíšete, bude připojeno k tomu, co již bylo načteno z vašeho souboru. Obvykle se odpovídá tak, že se z klávesnice zadá `\bye <CR>`<sup>1</sup>, protože tak se obvykle zpracovávání ukončí. Někdy se ovšem může přihodit, že vám chybí nějaká pravá závorka `}`, takže T<sub>E</sub>X odmítá i nadále skončit. V každém případě pomůže současné stisknutí kláves `CTRL+Z`.

#### 8.2 Překroucená a neznámá řídicí slova

Toto je také častá chyba. Je-li T<sub>E</sub>X spuštěn v dávkovém zpracování (batch mode), chybové hlášení se zobrazí na obrazovce a (chybná) řídicí sekvence je ignorována. Pracuje-li T<sub>E</sub>X interaktivně, je možné chyby napravit (to samozřejmě nemění původní vstupní soubor a opravu v něm je nutno provést až po skončení práce).

Mějme v T<sub>E</sub>Xu následující jednoduchý vstupní soubor, který se sestává ze dvou řádků:

---

<sup>1</sup> `<CR>` je klávesa sloužící k ukončení zadávání řádku. Mohli bychom ji nazvat návrat vozíku (carriage return nebo enter). Někdy je označena velkou šipkou vlevo.

```
\line{Levá strana \hfli pravá strana}  
\bye
```

Samozřejmě, že správné řídicí slovo mělo být `\hfil`. Na vaší obrazovce by se mělo objevit zhruba toto hlášení:

```
! Undefined control sequence.  
1.1 \line{ The left side \hfli  
                                the right side}  
?
```

neboli

```
! Nedefinovaná řídicí sekvence.  
1.1 \line{ Levá strana \hfli  
                                pravá strana}  
?
```

První řádek začíná vykřičníkem ! a obsahuje chybové hlášení. Potom následuje číslo řádku, ve kterém se chyba objevila, a ta část řádku, která byla přečtena úspěšně. Následující řádek je pokračováním řádku po chybě.

Označení chyby vykřičníkem a údaj o čísle řádku, kde se chyba vyskytla, má samozřejmě význam pro opravu chyb v souboru. S dobrým editorem nám to dokonce opravu jednotlivých chyb velmi usnadní. Malý pascalovský editor `te.exe`, který je dodáván v instalaci  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u, možná nevyhoví každému při psaní zdrojového textu, ale pokud váš soubor není příliš rozsáhlý (tj. není větší než 64 kB), umožňuje jednak současnou práci s oběma soubory `mujtext.tex` a `mujtext.log` a jednoduchý pohyb v obou oknech, ale hlavně stiskem jediné klávesy `<F8>` skok na řádku s chybou ve zdrojovém souboru, zatímco v druhém okně se objeví příslušná chybová hláška. Tato vlastnost je nakonfigurována i pro ostatní dva editory, které najdete v nabídce  $\mathcal{C}\mathcal{S}\mathcal{T}\mathcal{E}\mathcal{X}$ u (jedná se o `Qedit` a `\mathcal{C}\mathcal{S}\mathcal{E}\mathcal{D}`).

Nakonec se objeví otazník ?, který při zastavení programu znamená, že  $\mathcal{T}\mathcal{E}\mathcal{X}$  čeká na odpověď. Existuje několik vhodných odpovědí:

## Odpovědi na chybová hlášení T<sub>E</sub>Xu

Možná odpověď	Vstup v T <sub>E</sub> Xu	Výsledek
Help (Pomoc)	h <CR>	Důvody zastavení jsou vypisovány na obrazovku.
Insert (Vlož)	i <CR>	Následuje vložení opravy, která umožní další zpracování, ale neopravuje se vstupní soubor.
Exit (Ukončení)	x <CR>	Odchod z T <sub>E</sub> Xu. Úplné ukončené stránky se uloží do souboru s extenzí dvi.
Scroll (Roluj)	s <CR>	Výpis hlášení a pokračování po malých chybách.
Run (Běh)	r <CR>	Výpis hlášení a pokračování po každé chybě.
Quiet (Tíše)	q <CR>	Potlačení všech výpisů chyb na obrazovku.
Carry on (Pokračuj)	<CR>	T <sub>E</sub> X pokračuje automaticky, jak nejlépe umí.

V našem příkladu by rozumnou odpovědí mohlo být

```
h <CR>
```

a pomocí ní bychom dostali pomocné hlášení. Potom bychom vložili

```
i <CR>
```

kvůli úpravě textu. Na tomto místě T<sub>E</sub>X bude reagovat tím, že se na obrazovce objeví

```
insert>
```

a my provedeme opravu špatně zadaného řídicího slova na `\hfil` a odešleme pomocí `<CR>`). A zde je výsledek: na obrazovce se postupně objeví

```
? h
```

```
The control sequence at the end of the top line
of your error message was never \def'ed. If you have
misspelled it (e.g., '\hobx'), type 'I' and the correct
spelling (e.g., 'I\hbox'). Otherwise just continue,
and I'll forget about whatever was undefined.
```

```
? i insert>\hfil
```

```
[1]
```

neboli



? h

Řídicí sekvence na konci horního řádku chybového hlášení nebyla nikdy \def'ována. Jestliže jsi ji zkomolil (např. '\hobx'), stiskni 'I' a napiš ji správně (např. 'I \hbox'). Jinak pokračuj a já zapomenu na to, co bylo nedefinováno.

? i insert>\hfil  
[1]

Symbol [1] na konci znamená, že první (a jediná) stránka byla dokončena a odeslána do souboru \*.dvi. Původní vstupní soubor je samozřejmě třeba ještě opravit.

### 8.3 Špatně pojmenované písmo

Tato chyba je podobná zkomolení řídicí sekvence. Chybové hlášení je ale jiné a zpočátku trochu matoucí. Předpokládejme například, že ve vstupním souboru máte:

```
\font\sf = cmss01
```

V tomto případě jsou prohozeny číslice. Chybové a pomocné hlášení vypadají takto:

```
! Font \sf=cmss01 not loadable: Metric (TFM) file not found.  
<to be read again>  
                                \par  
\bye ->\par  
                                \vfill \supereject \end  
1.2 \bye
```

? h

I wasn't able to read the size data for this font,  
so I will ignore the font specification.  
[Wizards can fix TFM files using TFtoPL/PLtoTF.]  
You might try inserting a different font spec;  
e.g., type 'I\font<same font id>=<substitute font name>'.

neboli

```
! Font \sf=cmss01 nelze načíst: Metrický soubor (TFM) nenalezen.  
<nutno přečíst znovu>  
                                \par  
\bye ->\par
```

```
\vfill \supereject \end
1.2 \bye
```

? h

Nebyl jsem schopen přečíst rozměrová data pro tento font, takže budu ignorovat specifikaci fontu.

[Čarodějové mohou vyrobit soubory typu TFM pomocí TFtoPL/PLtoTF.]

Měl bys zkusit zadat jinou specifikaci fontu;

např., napsat 'I\font<tentýž font>=<náhradní jméno fontu>'.

Soubor s extenzí `tfm` (metrický soubor  $\TeX$ u) je pomocný soubor, který  $\TeX$  využívá. Toto podivné hlášení vám tedy říká, že definovaný font ve vašem počítači není dosažitelný. Vás už musí napadnout, jak chybu jednoduše opravit.

## 8.4 Popletená matematika

Jinou častou chybou je to, že se na začátek matematického výrazu napíše `$` nebo `$$`, ale pak se zapomene totéž napsat na konec výrazu. Text, který potom následuje, je chápán jako matematika; co je však ještě horší je to, že pokud začne další matematický text novým `$` nebo `$$`, bude chápán jako normální text. Rozumí se, že vznikne spousta chybových hlášení.  $\TeX$  se pokusí vyřešit tento problém vložím nového `$` nebo `$$`; v každém případě problém je vyřešen na konci odstavce, protože nový odstavec začíná automaticky jako obyčejný text.

Uvažujme následující správný vstupní text:

Protože  $f(x) > 0$ ,  $a < b$  a funkce  $f(x)$  je spojitá, platí  $\int_a^b f(x) dx > 0$ .

Na výstupu obdržíme

Protože  $f(x) > 0$ ,  $a < b$  a funkce  $f(x)$  je spojitá, platí  $\int_a^b f(x) dx > 0$ .

Vynecháme-li v  $f(x)$  druhý znak dolaru, dostaneme takovéto chybové a pomocné hlášení:

```
! Missing $ inserted.
<inserted text>
      $
<to be read again>
      \intop
\int ->\intop
      \nolimits
1.2 $\int
```

```

_a^b f(x)\,dx >0$.
? h
I've inserted a begin-math/end-math symbol since I think
you left one out. Proceed, with fingers crossed.

```

?

neboli

```

! Chybějící $ doplněn.
<vložený text>
      $
<nutno číst znovu>
      \intop
\int ->\intop
      \nolimits
1.2 $\int
      _a^b f(x)\,dx >0$.

```

```

? h
Doplnil jsem symbol začátku/konce matematického textu, protože si
myslím, že jsi jeden vynechal. Pokračuj a doufej, že to dobře
dopadne.

```

?

Řádek začínající ! říká, co se stalo. Řádek začínající 1.2 ukazuje, na kterém místě ve vstupním souboru se chyba objevila. Jak jsme již viděli v jiných příkladech, ta část řádku, která byla přečtena úspěšně, tj. až po `\int`, je na jednom řádku a zbytek pokračuje na dalším řádku. Konec hlášení se může zdát nejasný. Tato dílčí hlášení říkají, co se dělo ve střevech programu  $\text{T}_{\text{E}}\text{X}$ , když se objevila chyba. Začátečník je může ignorovat. A tady je výsledek po pokusu  $\text{T}_{\text{E}}\text{X}$ u chybu napravit:

Protože  $f(x) > 0$ ,  $a < b$  a funkce  $f(x)$  je spojitá, platí  $\int_a^b f(x) dx > 0$ .

Je to sražený text tištěný kurzívou a bez mezer. To je typické pro normální text, je-li chápán jako text matematický; jestliže uvidíte něco takového na výstupu, je skoro jisté, že jste vynechali `$` nebo `$$`.

## 8.5 Nesprávné závorkování

Při vytváření skupin snadno zapomeneme na uzavírací závorky nebo se nám popletou. Výsledkem může být zcela nepatrná chyba, ale někdy je situace katastrofální. Předpokládejme například, že v textu máme `\bf Tučný titulék`, a že jsme tedy omylem vynechali uzavírací složenou závorku. Výsledek bude stejný, ja-

ko by tam žádná otevírací složená závorka nebyla; jinými slovy, celý zbytek článku bude vytištěn tučně, nebudou-li již následovat žádné jiné změny fontu. Na konci souboru dostanete následující hlášení:

```
(\end occurred inside a group at level 1)
```

neboli

```
(\end se vyskytl ve skupině 1. úrovně)
```

Uděláte-li stejnou chybu dvakrát, pak bude o dvě otevírací složené závorky více než uzavíracích a měli byste dostat toto hlášení:

```
(\end occurred inside a group at level 2)
```

neboli

```
(\end se vyskytl ve skupině 2. úrovně)
```

$\TeX$  neví, že chybí uzavírací složená závorka do té doby, dokud nedojde na konec souboru. Hlášení vám tedy neřekne, kde jste udělali chybu. Není-li nalezení místa chybějící závorky snadné, je kdykoli možné vložit `\bye` doprostřed dokumentu. Spustíte-li  $\TeX$  znovu, bude se zpracovávat jen první polovina dokumentu, a jestliže chybová hlášení zůstanou stejná, potom víte, že chyba je v první polovině dokumentu. Přesouváním `\bye` na různá místa lze chybu lokalizovat. Podíváte-li se tedy na výstup několikrát, nakonec odhalíte, co a kde se stalo.

Chybějící otevírací složená závorka se rozpozná mnohem snadněji. Uvedme dvouřádkový vstupní soubor a výsledné chybové a pomocné hlášení:

```
\bf Každý začátek }má svůj konec.  
\bye
```

V hlášení se objeví

```
! Too many }'s.  
1.1 \bf Každý začátek}  
      má svůj konec.
```

```
? h  
You've closed more groups than you opened.  
Such booboos are generally harmless, so keep going.
```

neboli

```
! Příliš mnoho }.  
1.1 \bf Každý začátek}
```

má svůj konec.

? h

Uzavřel jsi více skupin než kolik jsi jich otevřel.

Taková bububu jsou obvykle neškodná, pokračuj tedy dál.

Samozřejmě se může stát, že chybějící levá složená závorka nebude na řádku, na kterém  $\TeX$  chybu zachytil.

Popletená složená závorka v definici nového řídicího slova může způsobit velkou chybu. Protože taková definice může obsahovat několik odstavců, nemusí být zachycena na konci odstavce a v neukončené definici se bude hromadit čím dál tím více textu. Může se dokonce stát, že se  $\TeX$ u nebude dostávat paměti, protože přidává další a další text. Tomu budeme říkat ujetá definice (runaway definition) — dobře to vystihuje situaci, i když to zní poněkud hovorově. Uveďme si dvouřádkový vstupní soubor s ujetou definicí:

$\TeX$ book:  
206

```
\def\noveslovo{def
\bye
```

Výsledné chybové a pomocné hlášení je toto:

```
Runaway definition?
->the def
! Forbidden control sequence found while scanning
definition of \noveslovo.
<inserted text>
      }
<to be read again>
      \bye
1.2 \bye

? h
I suspect you have forgotten a '}', causing me
to read past where you wanted me to stop.
I'll try to recover; but if the error is serious,
you'd better type 'E' or 'X' now and fix your file.
```

```
?
No pages of output.
```

neboli

Ujetá definice?

```

->the def
! Nalezena zakázaná řídicí sekvence při procházení definice
\noveslovo.
<vložený text>
    }
<nutno přečíst znovu>
    \bye
1.2 \bye

```

? h

Předpokládám, že jsi zapomněl na ‘}’, takže jsem přečetl více, než jsi zamýšlel.

Zkusím to napravit; je-li však chyba vážná, bude lepší, když teď stiskneš ‘E’ nebo ‘X’ a soubor upravíš.

?

Žádná stránka na výstupu.

To je samozřejmě závažná chyba. Objeví-li se na začátku souboru (jako v předchozím příkladu), nedostaneme na výstupu vůbec nic!

Vynechá-li se uzavírací složená závorka při použití makra s parametry, ujetá definice se ukončí na konci odstavce. Bylo-li definováno `\def\noveslovo#1{...}` a použijete-li `\noveslovo{...}` bez uzavírací závorky, pak se zničí nejvýše jeden odstavec.

TeXbook:  
205

Zkrátka a dobře: vyskytne-li se chyba, podívejte se na řádek začínající vykřičníkem a popisující chybu, potom se podívejte na řádek začínající číslem řádku, abyste se dozvěděli, kolik už bylo přečteno vstupního souboru TeXu. Je-li chyba stále ještě nejasná, zeptejte se TeXu pomocí `h <CR>` na další informace.

## Kapitola 9

### Kutání trochu hlouběji

---

V této kapitole se podíváme na několik prostředků, které umožňují používat  $\TeX$  pružněji nebo účinněji. Protože délka vytvářených dokumentů narůstá, je vhodné uvažovat o způsobech, jak jejich vytváření usnadnit.

#### 9.1 Velké soubory, malé soubory

$\TeX$  může číst a zapisovat soubory přímo za chodu. To umožňuje pracovat se soubory, které jsou menší, a proto je manipulace s nimi příjemnější. Například tento dokument se sestává z deseti kapitol a úvodu. Navíc existují makra, která se používají ve všech kapitolách. Tato makra lze uložit např. do souboru `makra.tex`, úvod do souboru `uvod.tex` a každou kapitolu do samostatného souboru. Řídící slovo `\input` potom zajistí načtení souboru. Obecně `\input jmenosbr` zajistí, že soubor `jmenosbr.tex` se načte a bude zpracováván ihned, stejně jako by text souboru `jmenosbr.tex` byl částí souboru, ve kterém byl načten. V tomto souboru se mohou načítat další soubory. Často je výhodné vytvořit samostatný soubor, který se čte postupně po malých částech, např.

```
\input makra
\input uvod
\input kap1
\input kap2
\input kap3
\input kap4
\input kap5
\input kap6
\input kap7
\input kap8
\input kap9
\input kap10
```

Je-li editace textu stále ještě obtížná, je možné zpracovávat jen některé soubory: stačí vložit `%` na začátek každého řádku souboru, který má být přeskočen.

Řídící slovo `\input` také umožňuje používat předdefinovaná makra. Je obtížné definovat, co to vlastně makro je. Zhruba řečeno, je to soubor definic, nastavení proměnných, instrukcí pro úpravu tisku atp. Makro pro psaní protokolů `proto.tex` by mohlo nastavit `\hsize`, `\vsize` a další parametry a také zajistit vytištění data

a časových údajů. Jakmile by toto bylo provedeno, každý protokol by mohl začínat řádkem `\input proto`, aby byl vtištěn v jednotně užívaném formátu.

▷ Cvičení 9.1 Vytvořte v  $\text{\TeX}$ u vstupní soubor `a.tex`, který bude načten v jiném souboru `b.tex`. Zkuste v souboru `b.tex` načíst soubor `a.tex` dvakrát pomocí dvojnásobného použití řídicího slova `\input`.

## 9.2 Větší makra

Navrhování maker, která lze použít při vytváření mnoha druhů dokumentů, je samozřejmě užitečné. Většina institucí a vysokých škol má požadavky na zvláštní a často komplikovaný formát pro vytváření preprintů nebo doktorských prací apod. Návrh kolekce maker, tj. balíku maker, která by splňovala všechny tyto požadavky, může být poněkud časově náročný. Můžete ho pak ale použít třeba i s jinými makry pomocí příkazu `\input`. Použití větších balíků maker umožňuje  $\text{\TeX}$  velmi snadným způsobem.

Balík maker lze uložit ve zvláštním tvaru, který  $\text{\TeX}$  dokáže rychle přečíst. Jsou to tzv. *formátové soubory* (format files), které mají extenzi `fmt` a jejichž přesný tvar je jen technickou záležitostí. Důležité je to, že umožňují, aby  $\text{\TeX}$  mohl pracovat s mnoha předem definovanými řídicími sekvencemi. Vždyť i standardní  $\text{\TeX}$  používá formátový soubor, který se nazývá `plain.fmt`.

Velmi je rozšířen balík maker  $\text{\LaTeX}$ . Tento balík umožňuje uživateli automaticky vytvořit rejstřík, obsah a seznam literatury. Zároveň umožňuje vkládání některých elementárních grafických obrazů jako jsou kružnice, ovály, přímky a šipky.  $\text{\LaTeX}$  také používá soubory stylů (style formats) k nastavení zvláštních parametrů stránky. K dispozici je řada různých souborů stylů; některé časopisy přijímají pro přímé zpracování příspěvky na magnetických médiích, jsou-li připraveny pomocí  $\text{\LaTeX}$ u a doporučeného stylu. Přejít od  $\text{\TeX}$ u k  $\text{\LaTeX}$ u není těžké. K dispozici je  $\text{\LaTeX}$ : A document preparation system.<sup>1</sup> Je to uživatelský manuál pro  $\text{\LaTeX}$ , který napsal autor tohoto makra Leslie Lamport.

Americká matematická společnost používá pro vydávání svých časopisů balík maker  $\text{\AMS-TeX}$ . Snadno ho lze od této společnosti získat a zasílat v  $\text{\AMS-TeX}$ u připravené příspěvky do jejích časopisů na magnetických médiích. Manuál pro psaní **The Joy of  $\text{\TeX}$**  od Michaela Spivaka lze získat také u Americké matematické společnosti.

Existují i jiné balíky maker a nepochybně budou vyvinuty další. Obvykle nestojí mnoho a za jistých okolností jsou velmi efektivní. Sdružení uživatelů  $\text{\TeX}$ u ( $\text{\TeX}$  Users Group) uveřejňuje ve svých publikacích informace o nových balících maker.

---

<sup>1</sup> Addison-Wesley Reading, Massachusetts, 1986, ISBN 0-201-15790-X.



### 9.3 Vodorovné a svislé linky

Kreslení vodorovných a svislých linek pomocí  $\TeX$ u je snadné. Napišeme-li v textu `\hrule`, potom se stávající odstavec ukončí, nakreslí se vodorovná linka s běžně nastavenou délkou (`hsize`) a pak se pokračuje novým odstavcem. Délku linky je možné nastavit, např. pomocí `\hrule width 5 cm`; zvětšení mezer nad linkou nebo pod ní lze zajistit pomocí `\vskip` nebo `\bigskip`. Např.

```
\parindent = 0 pt \parskip = 12 pt
Tento text je nad vodorovnou linkou (hrule)
\bigskip
\hrule width 3 in
a toto je text pod linkou.
```

dává

Tento text je nad vodorovnou linkou (hrule)



a toto je text pod linkou.

Tato linka má ve skutečnosti nejen délku tří palců, ale současně i výšku (výšku nad základní linkou, na kterou je nastavena sazba) 0,4 bodu a hloubku (pod základní linkou) 0 bodů. Každý z těchto parametrů lze nastavit zvlášť. Změníme-li čtvrtý řádek našeho příkladu takto: `\hrule width 3 in height 2 pt depth 3 pt`, dostaneme

Tento text je nad vodorovnou linkou (hrule)



a toto je text pod linkou.

Parametry `width` (délka), `height` (výška) a `depth` (hloubka) lze vypsát v libovolném pořadí.

Svislou linku lze definovat zcela podobně jako vodorovnou linku (zadáním parametrů `width`, `height` a `depth`, chcete-li). Avšak na rozdíl od vodorovné linky, svislá linka nezačíná v novém odstavci. Implicitní nastavení šířky je 0,4 bodu a výška linky je stejná jako výška řádku, do kterého se vysazuje text. Po zpracování textu

Tento text je před svislou linkou (vrule)

```
\vrule\
a toto je až za ní.
```

dostanete

Tento text je před svislou linkou (vrule) | a toto je až za ní.

▷ Cvičení 9.2 Vysázejte tři vodorovné linky, které jsou od sebe vzdáleny 15 bodů, jsou 3 palce dlouhé a jsou umístěny jeden palec od levého okraje.

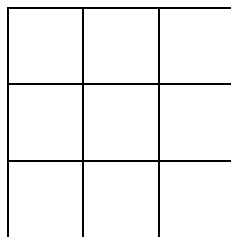
Ačkoliv si hrule (vodorovnou linku) a vrule (svislou linku) obvykle představujeme jako vodorovnou nebo svislou přímku, není nutné ji používat v tomto smyslu. Například

```
\noindent  
Jméno: \vrule height .4pt depth 0 pt width 3 in
```

dává

Jméno: \_\_\_\_\_

▷ Cvičení 9.3 Vysázejte následující mřížku (každý čtvereček má délku strany 1 cm):



## 9.4 Rámečky v rámečku

Jak jsme už viděli v diskuzi o tvarech linek, svislé a vodorovné boxy<sup>1</sup> (vbox, hbox) jsou objekty, které mohou být přeplněny nebo nenaplněny. V této sekci se budeme zabývat boxy trochu podrobněji. Pomocí boxů lze text umístit kdekoli na stránce.

Vodorovný box (hbox) se tvoří pomocí `\hbox{...}`. Jestliže to, co je mezi složenými závorkami, bylo vloženo do vodorovného boxu, potom je to stmeleno a nelze to dále dělit (jinými slovy, vloží-li se něco, co má být na jednom řádku, do vodorovného boxu, bude to tvořit celek). Rozměry vodorovného boxu lze předeepsat. Např. `\hbox to 5 cm{obsah rámečku}` dává box široký právě 5 cm a obsahující text „obsah rámečku“. Tímto způsobem lze ale snadno vyrobit přeplněné a nenaplněné boxy. Nejsou-li však rozměry boxu zadány, je hbox právě tak velký, aby se do něj jím zarámovaný text vešel.

---

<sup>1</sup> Omlouváme se Karlu Palovi za zavedení těchto alternativních termínů; po přečtení této sekce bude snad zřejmé, že je to téměř nutné.

Svislé boxy (vbox) se tvoří podobně pomocí `\vbox{...}`. Tyto boxy jsou zajímavé díky následující vlastnosti. Obsahuje-li vbox hboxy, potom se vodorovně řadí jeden pod druhým a tvoří jediný celek. Podobně hbox může obsahovat vboxy a ty se potom umístí do řádku — zarovnávají se podle základů. Vezmeme tři hboxy a vložíme je do vboxu:

```
\vbox{
  \hbox{Obsah rámečku č. 1}
  \hbox{Obsah rámečku č. 2}
  \hbox{Obsah rámečku č. 3}
}
```

(je vhodné si zvyknout tvořit vstupní text v co nejpřehlednější formě). Dostaneme tak

```
Obsah rámečku č. 1
Obsah rámečku č. 2
Obsah rámečku č. 3
```

Nyní vytvoříme další vbox:

```
\vbox{
  \hbox{Obsah rámečku č. 4}
  \hbox{Obsah rámečku č. 5}
}
```

Oba vboxy lze vložit do hboxu, v němž budou vedle sebe. Dostaneme tak konstrukci:

```
\hbox{
  \vbox{
    \hbox{Obsah rámečku č. 1}
    \hbox{Obsah rámečku č. 2}
    \hbox{Obsah rámečku č. 3}
  }
  \vbox{
    \hbox{Obsah rámečku č. 4}
    \hbox{Obsah rámečku č. 5}
  }
}
```

Po zpracování  $\TeX$ em dostaneme

Obsah rámečku č. 1  
Obsah rámečku č. 2    Obsah rámečku č. 4  
Obsah rámečku č. 3    Obsah rámečku č. 5

Všimněte si, že vboxy jsou opravdu zarovnaný tak, aby jejich základny byly na stejné lince. Mezi oběma vboxy je jen jedna mezera. Mezeru lze zvětšit např. o 1 cm vložením `\hskip 1 cm` mezi oba vboxy. Oba boxy lze také zarovnat podle jejich horních stran a to tak, že řídicí slovo `\vbox` nahradíme řídicím slovem `\vtop`. Po těchto dvou změnách obdržíme následující výsledek

Obsah rámečku č. 1            Obsah rámečku č. 4  
Obsah rámečku č. 2            Obsah rámečku č. 5  
Obsah rámečku č. 3

Vboxy, hboxy a linky můžeme kombinovat tak, abychom dostali zarámovaný text. Jak se to dělá? Jednou z možností je umístit text, který má být zarámován, mezi svislé linky, výsledek umístit do hboxu, umístit nad něj i pod něj vodorovnou linku a to všechno vložit do vboxu. Vypadá to takto:

```
\vbox {  
  \hrule  
  \hbox{\vrule Text, který má být zarámován \vrule }  
  \hrule  
}
```

Po zpracování dostaneme

Text, který má být zarámován

Dostali jsme sice zarámovaný text, ale zcela bez mezer kolem a to vypadá velmi křečovitě (T<sub>E</sub>X ovšem zplodil právě jen to, co jsme mu řekli). Situaci můžeme zlepšit, vložíme-li `\strut` na začátek hboxu, čímž se trochu zvětší. Tak dostaneme:

Text, který má být zarámován

▷ Cvičení 9.4 Pomocí metody zarámování umístěte text doprostřed boxu, který je široký přes celou stránku.

▷ Cvičení 9.5 Vytvořte následující magický čtverec:

6	1	8
7	5	3
2	9	4

▷ Cvičení 9.6 Navrhněte makro `\boxtext#1{...}`, které zarámuje text mezi složenými závorkami. Vyzkoušejte si své makro na větě s každým druhým slovem zarámovaným: Nejsm si zcela jist, že by tohle chtěl někdo dělat, protože nakonec se nedostane žádná lahůdka.

Všimněte si, jak jsou zarovnány základní linky řádků a základny rámečků.

Je docela snadné pohybovat rámečky na stránce nahoru, dolů, vlevo a vpravo. Posunout vbox o jeden palec vpravo lze pomocí `\moveright 1 in \vbox{...}`. K posunutí vlevo se použije `\moveleft`. Podobně lze posunout hbox nahoru nebo dolů pomocí řídicích slov `\raise` a `\lower`.

▷ Cvičení 9.7 Makro `\boxtext` z předcházejícího cvičení přepište tak, aby byl celý text zarovnán (implicitní nastavení hloubky `\strut` je 3,5 bodu). Měli byste dostat zhruba takovouto větu:

Nejsm si zcela jist, že by tohle chtěl někdo dělat, protože nakonec se nedostane žádná lahůdka.

Všimněte si, jak jsou zarovnány základní linky řádků a základny boxů.

Box lze vyplnit vodorovnými linkami nebo tečkami. Základní myšlenka spočívá v použití řídicích slov `\hrulefill` nebo `\dotfill` v boxu. Napišme do vstupního souboru

```
\hbox to 4.5 in{Začínáme\hrulefill 1}
\hbox to 4.5 in{Všechny velké a malé znaky\hrulefill 9}
\hbox to 4.5 in{Úprava věcí příštích\hrulefill 17}
\hbox to 4.5 in{Žádný strach z matematiky!\hrulefill 30}
```

Po zpracování dostaneme

Začínáme	1
Všechny velké a malé znaky	9
Úprava věcí příštích	17
Žádný strach z matematiky!	30

Zaměníme-li `\hrulefill` řídicím slovem `\dotfill`, dostaneme

Začínáme	1
Všechny velké a malé znaky	9
Úprava věcí příštích	17
Žádný strach z matematiky!	30

Nakonec ještě jednu poznámku: přečetli jste si úvodní text o používání  $\text{\TeX}$ . Pro běžné použití to možná bude stačit, ale zdaleka o něm nevíte vše. Tvorbou no-

vých maker se  $\text{\TeX}$  učí dělat stále komplikovanější věci. Již dnes jsou v Praze, Brně a Bratislavě (ale i leckde jinde) lidé, kteří se  $\text{\TeX}$ em intenzívně zabývají. Najdete je vcelku snadno a většina vám ráda pomůže, budete-li mít nějaké problémy. V době dokončování překladu této knížky se začali českoslovenští uživatelé  $\text{\TeX}$ u sdružovat při Jednotě československých matematiků a fyziků. Jak to dopadlo, je zřejmé z přidané kapitoly 11.

## Kapitola 10

### Seznam řídících znaků a slov

---

Následuje seznam všech řídících slov, o kterých byla zmínka v tomto manuálu. Chcete-li se o nich dozvědět něco víc, podívejte se do rejstříku v knize **The TeXbook**.

#### 10.1 Řídící znaky

<code>\_</code> , 27, 54	<code>\&amp;</code> , 30	<code>\;</code> , 54	<code>\{</code> , 30
<code>\!</code> , 54	<code>\'</code> , 31	<code>\=</code> , 31	<code>\}</code> , 30
<code>\"</code> , 31	<code>\+</code> , 65	<code>\&gt;</code> , 54	<code>\ </code> , 56, 60
<code>\#</code> , 30	<code>\,</code> , 54, 58	<code>\^</code> , 30, 31	<code>\~</code> , 30
<code>\\$</code> , 27, 30	<code>\.</code> , 31	<code>\-</code> , 30	
<code>\%</code> , 27, 30	<code>\/</code> , 36	<code>\'</code> , 31	

#### 10.2 Řídící slova

<code>\AA</code> , 32	<code>\biggl</code> , 59	<code>\columns</code> , 65
<code>\aa</code> , 32	<code>\Biggr</code> , 59	<code>\cos</code> , 60
<code>\acute</code> , 55	<code>\biggr</code> , 59	<code>\cosh</code> , 60
<code>\AE</code> , 32	<code>\Bigl</code> , 59	<code>\cot</code> , 60
<code>\ae</code> , 32	<code>\bigl</code> , 59	<code>\coth</code> , 60
<code>\aleph</code> , 56	<code>\Bigr</code> , 59	<code>\cr</code> , 62, 65
<code>\alpha</code> , 55	<code>\bigr</code> , 59	<code>\csc</code> , 60
<code>\angle</code> , 56	<code>\bigskip</code> , 42	<code>\cup</code> , 56
<code>\approx</code> , 56	<code>\boxtext</code> , 94	<code>\d</code> , 32
<code>\arccos</code> , 60	<code>\break</code> , 44	<code>\dag</code> , 46
<code>\arcsin</code> , 60	<code>\breve</code> , 55	<code>\ddag</code> , 46
<code>\arctan</code> , 60	<code>\bullet</code> , 56	<code>\ddot</code> , 55
<code>\arg</code> , 60	<code>\bye</code> , 26	<code>\def</code> , 74
<code>\ast</code> , 56	<code>\c</code> , 32	<code>\deg</code> , 60
<code>\b</code> , 32	<code>\cal</code> , 35	<code>\Delta</code> , 55
<code>\backslash</code> , 56	<code>\cap</code> , 56	<code>\delta</code> , 55
<code>\bar</code> , 55	<code>\cdot</code> , 56	<code>\det</code> , 60
<code>\baselineskip</code> , 41	<code>\centerline</code> , 44	<code>\diamond</code> , 56
<code>\beta</code> , 55	<code>\check</code> , 55	<code>\dim</code> , 60
<code>\bf</code> , 35	<code>\chi</code> , 55	<code>\div</code> , 56
<code>\Bigl</code> , 59	<code>\circ</code> , 56	<code>\dot</code> , 55

<code>\dotfill</code> , 66	<code>\i</code> , 31	<code>\nabla</code> , 56
<code>\dots</code> , 34	<code>\ifodd</code> , 46	<code>\narrower</code> , 42
<code>\Downarrow</code> , 60	<code>\Im</code> , 56	<code>\natural</code> , 56
<code>\downarrow</code> , 60	<code>\in</code> , 56	<code>\neg</code> , 56
<code>\eject</code> , 39	<code>\inf</code> , 60	<code>\ni</code> , 56
<code>\ell</code> , 56	<code>\infty</code> , 56	<code>\noindent</code> , 41
<code>\else</code> , 46	<code>\input</code> , 88	<code>\nopagenumbers</code> , 26, 39
<code>\emergencystretch</code> , 48	<code>\int</code> , 57	<code>\not</code> , 56
<code>\endinsert</code> , 42	<code>\iota</code> , 55	<code>\nu</code> , 55
<code>\epsilon</code> , 55	<code>\it</code> , 35	<code>\O</code> , 32
<code>\eqalign</code> , 63	<code>\item</code> , 44	<code>\o</code> , 32
<code>\eqno</code> , 64	<code>\itemitem</code> , 44	<code>\odot</code> , 56
<code>\equiv</code> , 56	<code>\j</code> , 31	<code>\OE</code> , 32
<code>\eta</code> , 55	<code>\kappa</code> , 55	<code>\oe</code> , 32
<code>\exists</code> , 56	<code>\ker</code> , 60	<code>\of</code> , 58
<code>\exp</code> , 60	<code>\L</code> , 32	<code>\offinterlineskip</code> , 72
<code>\flat</code> , 56	<code>\l</code> , 32	<code>\Omega</code> , 55
<code>\font</code> , 36	<code>\Lambda</code> , 55	<code>\omega</code> , 55
<code>\footline</code> , 46	<code>\lambda</code> , 55	<code>\ominus</code> , 56
<code>\footnote</code> , 46	<code>\langle</code> , 60	<code>\oplus</code> , 56
<code>\forall</code> , 56	<code>\lceil</code> , 60	<code>\otimes</code> , 56
<code>\Gamma</code> , 55	<code>\left</code> , 62	<code>\over</code> , 57
<code>\gamma</code> , 55	<code>\leftline</code> , 44	<code>\overfullrule</code> , 48
<code>\gcd</code> , 60	<code>\leftskip</code> , 42	<code>\overline</code> , 59
<code>\geq</code> , 56	<code>\leq</code> , 56	<code>\P</code> , 46
<code>\grave</code> , 55	<code>\leqno</code> , 64	<code>\pageno</code> , 46
<code>\H</code> , 32	<code>\let</code> , 78	<code>\par</code> , 28
<code>\halign</code> , 69	<code>\lfloor</code> , 60	<code>\parallel</code> , 56
<code>\hang</code> , 43	<code>\lg</code> , 60	<code>\parindent</code> , 41
<code>\hangafter</code> , 43	<code>\lim</code> , 58, 60	<code>\parshape</code> , 43
<code>\hangindent</code> , 43	<code>\liminf</code> , 60	<code>\parskip</code> , 41
<code>\hat</code> , 55	<code>\limsup</code> , 60	<code>\partial</code> , 56
<code>\hbadness</code> , 47	<code>\line</code> , 44	<code>\perp</code> , 56
<code>\hbox</code> , 91	<code>\ln</code> , 60	<code>\Phi</code> , 55
<code>\headline</code> , 46	<code>\log</code> , 60	<code>\phi</code> , 55
<code>\hfil</code> , 44, 45	<code>\lower</code> , 94	<code>\Pi</code> , 55
<code>\hfill</code> , 66	<code>\magnification</code> , 40	<code>\pi</code> , 55
<code>\hfilll</code> , 66	<code>\magstep</code> , 36	<code>\pmatrix</code> , 62
<code>\hfuzz</code> , 48	<code>\mathop</code> , 61	<code>\Pr</code> , 60
<code>\hoffset</code> , 40	<code>\matrix</code> , 62	<code>\proclaim</code> , 61
<code>\hom</code> , 60	<code>\max</code> , 60	<code>\Psi</code> , 55
<code>\hrule</code> , 71, 90	<code>\medskip</code> , 42	<code>\psi</code> , 55
<code>\hrulefill</code> , 66	<code>\min</code> , 60	<code>\qqquad</code> , 54
<code>\hsize</code> , 39, 90	<code>\moveleft</code> , 94	<code>\quad</code> , 54
<code>\hskip</code> , 45	<code>\moveright</code> , 72, 94	<code>\raggedright</code> , 45
<code>\hyphenation</code> , 48	<code>\mu</code> , 55	<code>\raise</code> , 94



<code>\rangle</code> , 60	<code>\strut</code> , 68	<code>\Updownarrow</code> , 60
<code>\rceil</code> , 60	<code>\subset</code> , 56	<code>\updownarrow</code> , 60
<code>\Re</code> , 56	<code>\subseteq</code> , 56	<code>\Upsilon</code> , 55
<code>\rfloor</code> , 60	<code>\sum</code> , 57	<code>\upsilon</code> , 55
<code>\rho</code> , 55	<code>\sup</code> , 60	<code>\v</code> , 32
<code>\right</code> , 62	<code>\supset</code> , 56	<code>\varepsilon</code> , 55
<code>\rightline</code> , 44	<code>\supseteq</code> , 56	<code>\varphi</code> , 55
<code>\rightskip</code> , 42	<code>\surd</code> , 58	<code>\varrho</code> , 55
<code>\rm</code> , 35	<code>\t</code> , 32	<code>\varsigma</code> , 55
<code>\root</code> , 58	<code>\tan</code> , 60	<code>\vartheta</code> , 55
<code>\S</code> , 46	<code>\tanh</code> , 60	<code>\vbadness</code> , 49
<code>\scaled</code> , 36	<code>\tau</code> , 55	<code>\vbox</code> , 92
<code>\sec</code> , 60	<code>\tenrm</code> , 46	<code>\vec</code> , 55
<code>\settabs</code> , 65	<code>\tensor</code> , 78	<code>\vee</code> , 56
<code>\sf</code> , 36	<code>\TeX</code> , 25	<code>\vfill</code> , 39
<code>\sharp</code> , 56	<code>\the</code> , 46	<code>\vglue</code> , 42
<code>\Sigma</code> , 55	<code>\Theta</code> , 55	<code>\voffset</code> , 40
<code>\sigma</code> , 55	<code>\theta</code> , 55	<code>\vrule</code> , 71
<code>\sim</code> , 56	<code>\tilde</code> , 55	<code>\vsize</code> , 39
<code>\simeq</code> , 56	<code>\times</code> , 56	<code>\vskip</code> , 90
<code>\sin</code> , 60	<code>\to</code> , 58	<code>\vtop</code> , 93
<code>\sinh</code> , 60	<code>\tolerance</code> , 48	<code>\wedge</code> , 56
<code>\sl</code> , 35	<code>\topinsert</code> , 42	<code>\widehat</code> , 55
<code>\smallskip</code> , 42	<code>\tt</code> , 35	<code>\widetilde</code> , 55
<code>\special</code> , 28	<code>\u</code> , 32	<code>\Xi</code> , 55
<code>\sqrt</code> , 58	<code>\underbar</code> , 59	<code>\xi</code> , 55
<code>\ss</code> , 32	<code>\underline</code> , 59	<code>\zeta</code> , 55
<code>\star</code> , 56	<code>\Uparrow</code> , 60	
<code>\stredlin</code> , 78	<code>\uparrow</code> , 60	

# Kapitola 11

## Několik závěrečných informací

---

V této části přinášíme několik informací o Československém sdružení uživatelů  $\text{T}_{\text{E}}\text{X}$ u ( $\text{C}_{\text{S}}\text{TUG}$ ). Tato organizace pečuje o popularizaci  $\text{T}_{\text{E}}\text{X}$ u v České a Slovenské republice, vydává svůj interní časopis, šíří bezplatně nebo za režijní cenu aktuální verze  $\text{T}_{\text{E}}\text{X}$ u (zejména pro PC) apod. Více napoví její stanovy:

### STANOVY

#### ČESKOSLOVENSKÉHO SDRUŽENÍ UŽIVATELŮ $\text{T}_{\text{E}}\text{X}$ u

##### Poslání Československého sdružení uživatelů $\text{T}_{\text{E}}\text{X}$ u

Československé sdružení uživatelů  $\text{T}_{\text{E}}\text{X}$ u je dobrovolným sdružením ve smyslu Zákona o sdružování občanů 83/1990 a jeho novely 68/1993 Sb.

Základním posláním Československého sdružení uživatelů  $\text{T}_{\text{E}}\text{X}$ u je vytvářet předpoklady pro všestranné využívání a další rozvoj jazyka počítačové typografie  $\text{T}_{\text{E}}\text{X}$  a příbuzného programového vybavení pro stolní tisk v Československu.

##### Základní ustanovení.

1. Československé sdružení uživatelů  $\text{T}_{\text{E}}\text{X}$ u je dobrovolná společnost uživatelů programového vybavení pro stolní tisk, a to zejména toho, které je založeno na jazyce  $\text{T}_{\text{E}}\text{X}$ . Působí na území Československé republiky.

2. Československé sdružení uživatelů  $\text{T}_{\text{E}}\text{X}$ u je samostatnou právnickou osobou se sídlem v Brně; sídlí na adrese Botanická 68a, 602 00 Brno.

3. V mezinárodní oblasti sdružení vystupuje pod názvem Czechoslovak  $\text{T}_{\text{E}}\text{X}$  Users Group a užívá zkráceného označení  $\text{C}_{\text{S}}\text{TUG}$ .

##### Činnost sdružení.

1. Sdružuje odborníky využívající programové vybavení pro stolní tisk (tzv. desktop publishing), založené zejména na využití jazyka  $\text{T}_{\text{E}}\text{X}$ . Sjednocuje jejich úsilí o efektivní propojení s typografickými podniky v Československu. Chrání zájmy uživatelů literatury využívající náročné technické sazby.

2. Spolupracuje s institucemi využívajícími výpočetní techniku v oblasti typografie, vydavatelstvími odborných časopisů a odbornými společnostmi, zejména Jednotou československých matematiků a fyziků a Československou informatickou společností.

3. Spolupracuje s obdobně zaměřenými společnostmi vytvářenými na základě územní či jazykové jednoty a společností TeX Users Group se sídlem v USA s celosvětovou působností.

4. Pořádá školení, diskuse, exkurze, konference, sympozia a semináře. Vyvíjí expertní, publikační, vydavatelskou a další odbornou činnost v oboru náročné technické sazby a DTP. Produkuje nebo zadává vytváření programového vybavení pro oblast využívání TeXu a toto vybavení udržuje a distribuuje členům.

5. Podporuje vytváření archivů softwaru (zejména TeXware) dostupných prostřednictvím sítě a jejich udržování.

6. Zajišťuje základní administrativu sdružení případně prostřednictvím síly či sil najímaných za úplatu.

### **Členství v Československém sdružení uživatelů TeXu, práva a povinnosti členů.**

1. Československé sdružení uživatelů TeXu má členy řádné a členy kolektivní z České republiky i ze zahraničí, zejména ze Slovenské republiky

2. Řádným členem sdružení se může stát každý občan či pracovník, který se přihlásí k myšlence poslání sdružení. S členstvím je neslučitelné porušování autorských práv a jiných zákonných a etických norem nakládání s duševním vlastnictvím.

3. Kolektivním členem sdružení se může stát každá instituce nebo podnik se zájmem o využívání typografie založené na použití TeXu, pokud se ztotožní s posláním sdružení.

4. Řádní i kolektivní členové jsou přijímáni usnesením výboru sdružení na základě přihlášky specifikující druh členství.

5. Řádným členem je každý, kdo byl jako řádný člen přijat a plní řádně své členské povinnosti. Řádný člen má právo volit a být volen do orgánů sdružení. Za kolektivního člena má právo volit jeho pověřený zástupce s váhou 3 hlasy.

6. Členství v Československém sdružení uživatelů TeXu zaniká vystoupením nebo vyloučením člena. Vyloučit člena může výbor sdružení, jestliže pro toto rozhodnutí hlasuje aspoň polovina jeho členů, a to z vážných důvodů, zejména pro dlouhodobé neplnění povinností člena sdružení.

7. Všichni členové sdružení mají právo být pravidelně informováni o činnosti orgánů sdružení. Informace jsou šířeny převážně prostřednictvím sítě Internet, jsou publikovány ve Zpravodaji sdružení a v závažných případech rozesílány poštou.

8. Všichni členové mají povinnost dodržovat stanovy a platit ve stanovené lhůtě příspěvky, jejichž výši určuje výbor sdružení.

### **Organizační struktura sdružení.**

1. Nejvyšším orgánem sdružení je valné shromáždění všech řádných členů.

2. Výkonným orgánem sdružení je výbor volený v počtu 10 – 15 členů. Výbor je volen a odvoláván valným shromážděním sdružení, a to nadpoloviční většinou. Volby mohou být organizovány korespondenčně. Výbor musí svolávat valné shromáždění alespoň jednou ročně v příslušném kalendářním roce s přesahem nejvýše dvou měsíců do dalšího roku. Funkční období voleného výboru je nejvýše tříleté.

3. Výbor sdružení volí ze svého středu předsedu. Pověřuje jednotlivé své členy vybranými úkoly tak, aby

- (a) byla zajištěna operativní a efektivní činnost sdružení,
- (b) bylo respektováno státoprávní uspořádání.

4. Ve funkčním období výboru je možno rozhodnutím výboru obsazovat funkci předsedy na kratší období, nejméně však 1 rok. Nemůže-li předseda ze závažných důvodů dočasně vykonávat své povinnosti, musí pověřit se souhlasem výboru zastupováním jiného člena výboru a toto oznámit členům sdružení. Funkce člena výboru není honorována, členům výboru lze proplácet nutné náklady spojené např. s cestou na zasedání apod.

5. Výbor řídí sdružení v období mezi valnými shromážděními a za svou činnost je odpovědný valnému shromáždění. V období mezi schůzemi výboru řídí činnost sdružení předseda výboru. Jedenkrát ročně podává členům informaci o hospodaření sdružení a o rámcových plánech činnosti. Výbor může z naléhavých důvodů kooptovat další řádné členy sdružení. Toto je povinen oznámit na nejbližším valném shromáždění.

6. Za Československé sdružení uživatelů TeXu jedná předseda nebo jiný výborem pověřený člen sdružení. O svých rozhodnutích podává členstvu zprávu ve Zpravodaji sdružení, zpravidla formou sdělení předsedy. Ze zasedání výboru je nutno pořídit zápis, který podepisuje člen výboru pověřený jeho zpracováním a autorizuje předseda.

7. V rámci sdružení pracují pracovní skupiny zřízované výborem, které sdružují členy podle specifických zájmů, resp. na regionálním principu. Tyto skupiny se řídí organizačním řádem schvalovaným výkonným výborem.

8. K řešení specifických problémů jsou výborem jmenovány komise. Sdružení dle potřeby zaměstnává k výkonu administrativních prací apod. pracovníky na základě smlouvy, kterou schvaluje výbor sdružení.

9. Kontrolními orgány sdružení jsou revizoři volení valným shromážděním.

10. O závažných otázkách valné shromáždění hlasuje. Právo hlasovat mají řádní členové a pověření zástupci kolektivních členů, jejichž hlas má váhu 3. Není-li výslovně stanoveno jinak, rozhoduje 30 min. po zahájení řádně svolané schůze prostá většina hlasů. Jeho rozhodnutí jsou pro výbor sdružení závazná.

11. K řešení sporů uvnitř sdružení volí valné shromáždění rozhodčí komisi, která je podřízena přímo valnému shromáždění sdružení.

### **Hospodaření sdružení.**

Činnost sdružení je hospodářsky zajišťována

- (a) z příspěvků členů,
- (b) z výtěžku služeb poskytovaných sdružením,
- (c) z darů a odkazů.

1. Za finanční hospodaření sdružení zodpovídá výbor sdružení, dispoziční právo s účty má předseda sdružení. Výbor zpravidla pověřuje svého člena (hospodáře), aby pomáhal předsedovi s hospodářskou agendou a případně s řízením práce kvalifikované externí síly, pověřené péčí o účetní doklady sdružení. Ve věcech finančních zodpovídá výboru za využívání bankovních účtů předseda sdružení, a to vždy v příslušném funkčním období.

2. Zpráva o hospodaření se předkládá pravidelně jednou ročně členstvu, zpravidla na valném shromáždění sdružení.

3. Výši příspěvků v dalším roce určuje výbor s přihlédnutím k vývoji cen vždy do 30. listopadu kalendářního roku. Příspěvky na kalendářní rok je nutno zaplatit vždy do 30. června tohoto roku. Nezaplatí-li člen bez vážné příčiny ani po následné urgenci, může být rozhodnutím výboru ze sdružení vyloučen. Pokud výbor změnu výše příspěvků v dané lhůtě neprojedná, výše příspěvků se nemění.

### **Závěrečná ustanovení.**

1. Stanovy Československého sdružení uživatelů  $\text{T}_{\text{E}}\text{X}$ u mohou být měněny pouze valným shromážděním sdružení, jestliže se pro změnu vysloví více než polovina odevzdaných hlasů.

2. Sdružení zaniká usnesením valného shromáždění sdružení, vysloví-li se pro tento návrh aspoň dvě třetiny odevzdaných hlasů. Všechny majetek, který zůstane po likvidaci, bude odevzdán orgánům určeným usnesením valného shromáždění o likvidaci.

3. Tato verze stanov navazuje na stanov sdružení registrované 9. května 1990; změny stanov sdružení byly schváleny valným shromážděním sdružení dne 25. května 1996 a výbor byl pověřen jejich dopracováním. Toto bylo provedeno ke dni 1. 7. 1996.

### **Příloha ke stanovám: Specifikace práv a povinností členů CSTUG**

#### **Práva řádného člena CSTUG.**

1. Dostávat zpravidla čtyřikrát ročně Zpravodaj sdružení.
2. Na vyžádání získávat veřejně dostupný software a software šířený v rámci CSTUG prostřednictvím sítě či z nejbližšího lokálního centra. Tento software je poskytován včetně nových verzí. Manipulační poplatky spojené s šířením (kopírování, rozesílání apod.) se určují dohodou.
3. Možnost publikovat články týkající se  $\text{T}_{\text{E}}\text{X}$ u ve Zpravodaji.
4. Přednostní právo v účasti na konferencích o  $\text{T}_{\text{E}}\text{X}$ u organizovaných sdružením.

5. Hlasovací právo na valném shromáždění CSTUG.
6. Být volen do výboru CSTUG a do funkce revizora CSTUG.

#### **Povinnosti řádného člena CSTUG.**

1. Platit členský příspěvek. V roce 1999 jsou příspěvky stanoveny takto: 260,- Kč (student: 160,- Kč).
2. Informovat správce databáze členů CSTUG nebo výbor CSTUG o všech změnách tak, aby údaje v databázi byly neustále aktuální.
3. Informovat o získaných důležitých informacích (odjinud než z CSTUG) výbor a na požádání výboru prostřednictvím Zpravodaje i ostatní členy (např. recenze literatury, softwaru, nový přístup ke zdrojům softwaru atp.).
4. Propagovat  $\TeX$  při všech příležitostech a získávat aktivně další řádné i kolektivní členy CSTUG.
5. Pomáhat podle svých možností ve svém okolí řešit problémy související s  $\TeX$ em.
6. Software vyvinutý či zakoupený pro CSTUG, který není označen v popisu jako veřejně dostupný, nesmí bez souhlasu výboru poskytnout uživatelům, kteří nejsou členy CSTUG.

#### **Práva kolektivního člena CSTUG.**

1. Dostávat čtyřikrát ročně informační bulletin ve třech exemplářích.
2. Na vyžádání získávat public domain software a software šířený v rámci CSTUG z nejbližšího lokálního centra a používat jej na všech svých pracovištích. Tento software je poskytován včetně nových verzí za režijní cenu za kopírování.
3. Zaměstnanci kolektivního člena mají právo 3) řádného člena CSTUG.

#### **Povinnosti kolektivního člena CSTUG.**

1. Platit členský příspěvek. V roce 1999 jsou příspěvky stanoveny takto: 1500,- Kč (střední škola: 500,- Kč).
2. Dále má kolektivní člen povinnosti 2) a 3) řádného člena
3. Pracovníky (případně studenty) instituce, která je kolektivním členem CSTUG, poučit o tom, že software vyvinutý či zakoupený pro CSTUG, který není označen v popisu jako veřejně dostupný, nesmí bez souhlasu výboru poskytnout uživatelům, kteří nejsou členy CSTUG nebo nepracují na pracovišti kolektivního člena CSTUG.

V Praze dne 25. května 1996

## Vybrané elektronické adresy

Adresa konference o českém a slovenském T<sub>E</sub>Xu ..... `cstex@cs.felk.cvut.cz`  
Přihláška do této konference (`subscribe cstex`) .. `listserv@cs.felk.cvut.cz`  
Reklamace, připomínky a návrhy k C<sub>S</sub>T<sub>E</sub>Xu ..... `cstexerr@cs.felk.cvut.cz`

Některé veřejné archívy T<sub>E</sub>Xovského software.

ČVUT v Praze ..... `cs.felk.cvut.cz` (192.108.160.1)  
Masarykova univerzita v Brně ..... `ftp.muni.cz` (147.251.12.8)  
Na Astonské universitě v Birminghamu ..... `ftp.tex.ac.uk` (134.151.79.32)  
Ve Stuttgartu ..... `ftp.uni-stuttgart.de` (129.69.8.13)  
Švýcarsko ..... `nic.switch.ch` (130.59.1.40)  
USA, Sam Houston State University ..... `niord.shsu.edu` (192.92.115.8)

Adresy některých členů současného výboru CSTUGu.

Martin Bílý ..... `bily@cs.felk.cvut.cz`  
Miroslav Dont ..... `dont@math.feld.cvut.cz`  
Karel Horák ..... `horakk@csearn.bitnet`  
Ladislav Lhotka ..... `lada@entu.cas.cz`  
Karol Nemoga ..... `matenemo@savba.savba.sk`  
Petr Novotný ..... `novotnyp@csearn.bitnet`  
Petr Olšák ..... `olsak@math.feld.cvut.cz`  
Stefan Porubský ..... `porubskvs@vscht.cz`  
Petr Sojka ..... `sojka@muni.cz`  
Oldřich Ulrych ..... `ulrych@karlin.mff.cuni.cz`  
Jiří Veselý ..... `jvesely@karlin.mff.cuni.cz`  
Zdeněk Wagner ..... `wagner@earn.cvut.cz`

## Kapitola 12

### Vyřešeno s malou pomocí

---

V této části nalezne čtenář řešení **některých** cvičení. Odpovědi nejsou číslovány — není to jen výrazem nechuti překladatelů upravit text podle poslední verze autora (tam ostatně je volena stejná úprava); mělo by to čtenáři vnuknout myšlenku, že v některých případech je skutečně lehčí odpověď vymyslet nežli vyhledat. V mnoha cvičeních není odpověď jednoznačně určena. Bude-li se vám líbit víc vaše řešení, rozhodně ho použijte!

Do překladu jsme zařadili i některé příklady, které se v textu nevyskytly. Některé jsou převzaty z poslední verze Doobova textu, jiné jsme považovali za užitečnou pomoc čtenáři, který se s  $\TeX$ em setkává poprvé. Částečně z technických důvodů a také s ohledem na možný výskyt „ $\TeX$ ářů-puristů“ jsou textové části uváděny v  $\TeX$ ové podobě.

---

Toto je moje první filologicky  
nenáročná věta  
v  $\TeX$  u.  
Byl jsem to já, kdo ji vysázel!

Toto je moje první filologicky nenáročná věta v  $\TeX$ u. Byl jsem to já, kdo ji vysázel!

---

---

Gratuluji! Při poslední zkoušce jste na 100\% vyhověl.

Gratuluji! Při poslední zkoušce jste na 100% vyhověl.

---

---

Dlužíte mi  $\$10.00$  a je na čase, abyste mi je poslal!

Dlužíte mi \$10.00 a je na čase, abyste mi je poslal!

---

---

Mám rád  $\TeX$  a stále ho používám. Bude se líbit i vám. Jakmile s ním začnete zacházet, je jeho užívání opravdu jednoduché. Stačí jen zvládnout  $\TeX$ nické detaily.

Mám rád  $\TeX$  a stále ho používám. Bude se líbit i vám. Jakmile s ním začnete zacházet, je jeho užívání opravdu jednoduché. Stačí jen zvládnout  $\TeX$ nické detaily.

---



---

Rozumí \AE schylus \OE dipovi?

Rozumí Æschylus Œdipovi?

---

---

Nejmenší vnitřní jednotka délky

v<sup>~</sup>\TeX{}u je asi 53.63\AA.

Nejmenší vnitřní jednotka délky v T<sub>E</sub>Xu je asi 53.63Å.

---

---

Plus \c ca change, plus \c ca la m<sup>^</sup>eme chose.

Plus ça change, plus ça la même chose.

---

---

\'El\'eves, refusez vos le\c cons! Jetez vos cha<sup>^</sup>\i nes!

Élèves, refusez vos leçons! Jetez vos chaînes!

---

---

Za\v sto tako polako pijete \v caj?

Zašto tako polako pijete čaj?

---

---

Ich mu\ss\ hei\ss en Tee abk\"uhlen.

Ich muß heißen Tee abkühlen.

---

---

Maar die \"ys is lekker.

Maar die ÿs is lekker.

---

---

Peut-<sup>^</sup>etre il pr\'ef\'ere le caf\'e glac\'e.

Peut-être il préfère le café glacé.

---

---

Lze jet trajektem z<sup>~</sup>\\"Olandu do \AA landu?

Lze jet trajektem z Ölandu do Ålandu?

---

---

Zpívala-li tra-la-la, ráda k<sup>~</sup>tomu --- poněkud

neobratně --- poskakovala.

Zpívala-li tra-la-la, ráda k tomu — poněkud neobratně — poskakovala.

---

---

Zima 1942--1943 byla nejhorší --- mnoho lidí nepřežilo.

Zima 1942–1943 byla nejhorší — mnoho lidí nepřežilo.

---

---

\uv{Jeho jasnost}, ohlásil komoří.

„Jeho jasnost“, ohlásil komoří.

---

---

Jaroslav vyhrkl: \uv{Opravdu je taková, že neumí říci {'ne'}?}

Jaroslav vyhrkl: „Opravdu je taková, že neumí říci ‘ne’?“

---

---

\uv{\dots až na věky věkův \dots a~nikdy jinak.}

„... až na věky věkův ... a nikdy jinak.“

---

---

Mr. J. B. Smith z Washingtonu, D. C., přijede z U. S. A. do \v CSFR v pátek 13. 8. v 9. 30.

Mr. J. B. Smith z Washingtonu, D. C., přijede z U. S. A. do ČSFR v pátek 13. 8. v 9. 30.

---

---

Nezkracujte pravou délku úsečky následovně:

pr. d. úsečky; raději pište

pr. dél. úsečky nebo

ještě lépe pr. délka úsečky.

(Prof. K. Havlíček.)

Nezkracujte pravou délku úsečky následovně: pr. d. úsečky; raději pište pr. dél. úsečky nebo ještě lépe pr. délka úsečky. (Prof. K. Havlíček.)

---

---

\medskip

\line{levé křídlo \hfil levý útočník

\hfil střední útočník \hfil pravý útočník

\hfil pravé křídlo}

levé křídlo    levý útočník    střední útočník    pravý útočník    pravé křídlo

\line{0 \hfil \hfil \hfil 3/5 \hfil 4/5 \hfil 1}

0                                  3/5                                  4/5                                  1

\medskip\line{\hskip 1~in JEDNA \hfil DVA \hfil TŘI}

JEDNA

DVA

TŘI

$C(n, r) = n! / (r! (n - r)!)$

$C(n, r) = n! / (r! (n - r)!)$

$a + b = c - d = xy = w/z$

$a + b = c - d = xy = w/z$

$a + b = c - d = xy = w/z$

$a + b = c - d = xy = w/z$

$f(x, y) = x' + x(x + y)$

$f(x, y) = x' + x(x + y)$

$f(x, y) = x' + x(x + y)$

$f(x, y) = x' + x(x + y)$

$\alpha\beta = \gamma + \delta$

$\alpha\beta = \gamma + \delta$

$\alpha\beta = \gamma + \delta$

$\alpha\beta = \gamma + \delta$

$\Gamma(n) = (n - 1)!$

$\Gamma(n) = (n - 1)!$

$\Gamma(n) = (n - 1)!$

$\Gamma(n) = (n - 1)!$

---

$\vec{x} \cdot \vec{y} = \langle \vec{x}, \vec{y} \rangle = 0$ ,  
 $\vec{x} \perp \vec{y}$ ,  
 právě když  $\vec{x} \perp \vec{y}$   
 $\vec{x} \cdot \vec{y} = \langle \vec{x}, \vec{y} \rangle = 0$ , právě když  $\vec{x} \perp \vec{y}$

---

$\vec{x} \cdot \vec{y} = \langle \vec{x}, \vec{y} \rangle \neq 0$ ,  
 $\vec{x} \not\perp \vec{y}$ ,  
 právě když  $\vec{x} \not\perp \vec{y}$

---

$(\forall x \in \mathbb{R})(\exists y \in \mathbb{R}) y > x$ .  
 $(\forall x \in \mathbb{R})(\exists y \in \mathbb{R}) y > x$ .

---

$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$   
 $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$

---

$2+4+6+\dots+2n = n(n+1)$   
 $2 + 4 + 6 + \dots + 2n = n(n + 1)$

---

$e^x \quad e^{-x} \quad e^{i\pi} + 1 = 0 \quad x_0 \quad x_0^2 \quad x_0^2 \quad 2^{x^x}$ .  
 $e^x \quad e^{-x} \quad e^{i\pi} + 1 = 0 \quad x_0 \quad x_0^2 \quad x_0^2 \quad 2^{x^x}$ .

---

$\nabla^2 f(x,y) = \{\partial^2 f \over \partial x^2\} + \{\partial^2 f \over \partial y^2\}$ .  
 $\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$ .

---

$\lim_{x \rightarrow 0} (1+x)^{\frac{1}{x}} = e$ .  
 $\lim_{x \rightarrow 0} (1+x)^{\frac{1}{x}} = e$ .

---

---

Mohutnost množiny  $(-\infty, \infty)$  je  $\aleph_1$ .

Mohutnost množiny  $(-\infty, \infty)$  je  $\aleph_1$ .

---

---

$\int_0^1 3x^2 dx = 1$ .

$$\int_0^1 3x^2 dx = 1.$$

---

---

$\sqrt{2} \quad \sqrt{\frac{x+y}{x-y}} \quad \sqrt[3]{10} \quad e^{\sqrt{x}}$

$$\sqrt{2} \quad \sqrt{\frac{x+y}{x-y}} \quad \sqrt[3]{10} \quad e^{\sqrt{x}}$$

---

---

$\frac{a+b}{c} \quad \frac{a}{b+c} \quad \frac{1}{a+b+c} \neq \frac{1}{a} + \frac{1}{b} + \frac{1}{c}$

$$\frac{a+b}{c} \quad \frac{a}{b+c} \quad \frac{1}{a+b+c} \neq \frac{1}{a} + \frac{1}{b} + \frac{1}{c}$$

---

---

$\|x\| = \sqrt{x \cdot x}$ .

$$\|x\| = \sqrt{x \cdot x}$$

---

---

$\underline{x} \quad \overline{y} \quad \underline{\overline{x+y}}$ .

$$\underline{x} \quad \overline{y} \quad \underline{\overline{x+y}}$$

---

---

$\lceil x \rceil \quad \lfloor x \rfloor \quad \lceil \lfloor x \rfloor \rceil \quad \lfloor \lceil x \rceil \rfloor$

$$\lceil x \rceil \leq \lfloor \lceil x \rceil \rfloor$$

---

---

$\sin(2\theta) = 2\sin\theta\cos\theta$   
 $\cos(2\theta) = 2\cos^2\theta - 1$

$$\sin(2\theta) = 2\sin\theta\cos\theta \quad \cos(2\theta) = 2\cos^2\theta - 1$$

---

---

$\int \csc^2 x dx = -\cot x + C$

$$\lim_{\alpha \rightarrow 0} \frac{\sin\alpha}{\alpha} = 1$$

$\lim_{\alpha \rightarrow \infty} \frac{\sin \alpha}{\alpha} = 0.$

$$\int \csc^2 x \, dx = -\cot x + C \quad \lim_{\alpha \rightarrow 0} \frac{\sin \alpha}{\alpha} = 1 \quad \lim_{\alpha \rightarrow \infty} \frac{\sin \alpha}{\alpha} = 0.$$

---

---

$\tan(2\theta) = \frac{2 \tan \theta}{1 - \tan^2 \theta}.$

$$\tan(2\theta) = \frac{2 \tan \theta}{1 - \tan^2 \theta}.$$

---

---

**Tvrzení 1.**

$\sum_{i < j}^n |X_i - X_j| = 0$ , právě když  $X_1 = \dots = X_n$ .

**Tvrzení 1.**  $\sum_{i < j}^n |X_i - X_j| = 0$ , právě když  $X_1 = \dots = X_n$ .

---

---

$I_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$

$$I_4 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

---

---

$|x| = \begin{cases} x & x \geq 0 \\ -x & x \leq 0 \end{cases}$

$$|x| = \begin{cases} x & x \geq 0 \\ -x & x \leq 0 \end{cases}$$

---

---

$\text{\settabs} \text{\+} \text{\hspace 2 in} \& \text{\hspace .75in} \& \text{\hspace 1cm} \text{\cr}$   
 $\text{\+} \& \text{\Švestky} \& \text{\hfill 8\,00 Kč} \text{\cr}$   
 $\text{\+} \& \text{\Káva} \& \text{\hfill 12\,00 Kč} \text{\cr}$   
 $\text{\+} \& \text{\Žvýkačka} \& \text{\hfill 10\,00 Kč} \text{\cr}$   
 $\text{\+} \& \text{\Houby} \& \text{\hfill 16\,50 Kč} \text{\cr}$   
 $\text{\+} \& \text{\Utopenec} \& \text{\hfill 12\,00 Kč} \text{\cr}$

```

\+ &Pomeranče &\hfill 16&,40 Kč\cr
\+ &Slivovice &\hfill 80&,00 Kč\cr
\+ &Rohlík &\hfill 0&,30 Kč\cr
\+ &Cigarety &\hfill 90&,00 Kč\cr
\+ &Uzený bůček &\hfill 18&,60 Kč\cr
\+ &Pivo &\hfill\underbar{\ \ 35}&\underbar{,00 Kč} \cr
\+ &Celkem &\hfill 298&,80 Kč\cr

```

Švestky	8,00 Kč
Káva	12,00 Kč
Žvýkačka	10,00 Kč
Houby	16,50 Kč
Utopenec	12,00 Kč
Pomeranče	16,40 Kč
Slivovice	80,00 Kč
Rohlík	0,30 Kč
Cigarety	90,00 Kč
Uzený bůček	18,60 Kč
Pivo	<u>35,00 Kč</u>
Celkem	298,80 Kč

---

```

\medskip
\settabs \+ \hskip 1cm&\hskip 1 cm&\hskip 1 cm& \cr
\moveright 2 in
\boxed{
\hrule width 3 cm
\+ \vrule height 1 cm & \vrule height 1 cm & \vrule height 1 cm
& \vrule height 1 cm \cr
\hrule width 3 cm
\+ \vrule height 1 cm & \vrule height 1 cm & \vrule height 1 cm
& \vrule height 1 cm \cr
\hrule width 3 cm
\+ \vrule height 1 cm & \vrule height 1 cm & \vrule height 1 cm
& \vrule height 1 cm \cr
\hrule width 3 cm
}

```


---

<code>\hbox to 4.5 in{Začínáme\dotfill 1}</code>	
<code>\hbox to 4.5 in{Všechny velké a ~malé znaky\dotfill 9}</code>	
Začínáme .....	1
Všechny velké a malé znaky .....	9

---

```

\def\boxtext#1{%
\ vbox{%
\hrule
\hbox{\strut \vrule{ } #1 \vrule}%
\hrule
}%
}
\moveright 2 in \vbox{\offinterlineskip
\hbox{\boxtext{6}\boxtext{1}\boxtext {8}}
\hbox{\boxtext{7}\boxtext{5}\boxtext{3}}
\hbox{\boxtext{2}\boxtext{9}\boxtext{4}}
}

```

6	1	8
7	5	3
2	9	4

---