UNIVERZITA PALACKÉHO V OLOMOUCI
PŘÍRODOVĚDECKÁ FAKULTA
KATEDRA MATEMATICKÉ ANALÝZY A APLIKACÍ MATEMATIKY

BAKALÁŘSKÁ PRÁCE

Databáze v TEXu
(Zpracování výsledků soutěže)

Vedoucí bakalářské práce:
RNDr. Miloslav Závodný
Rok odevzdání: 2012

Vypracoval:
Tereza Šolcová
ME, III. ročník

Prohlášení

Prohlašuji, že jsem bakalářskou práci zpracovala samostatně pod vedením pana RNDr. Miloslava Závodného s použitím uvedené literatury.

V Olomouci dne 10. dubna 2012

Poděkování

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

Obsah

Úvod

Téma mé bakalářské práce je *Databáze v TEXu*. Důvodem, proč jsem si vybrala toto téma bylo, že na naší katedře jsem měla možnost nahlédnout do textů sázených programem TEX a způsob jejich zpracování se mi velice zalíbil. Výsledek byl úhledný a já jsem chtěla, aby mé práce vypadaly také tak, proto jsem se rozhodla naučit se pracovat se systémem TEX. Zaujala mě pak možnost programovat TEX a vytvářet si skripty řešící hromadné zpracování dokumentů, což se mi jeví pro praxi jako velmi užitečné.

Cílem bakalářské práce je vytvořit textovou databázi bez databázového programu a ukázat její použití na přípravě „textového ocenění“ (tisk diplomů) vhodné soutěže.

Druhým cílem mé práce je zpracovat dané téma natolik srozumitelně, aby posloužila případným zájemcům jako učební pomůcka. Publikace na toto téma totiž nejsou hojně rozšířené, jednou z mála výjimek je článek [4], který mi posloužil jako základní pramen při studiu zadané problematiky.

Práci jsem rozdělila do 4 kapitol. V první kapitole nalezneme stručné seznámení s programem TEX. Druhá kapitola se zabývá teoretickým úvodem do databází. Ve třetí kapitole si ukážeme jednoduchý příklad a podrobně si ho vysvětlíme. Ke konci kapitoly je uvedena zmínka o podmíněných příkazech (větvení, if příkazech). Ve čtvrté kapitole si popíšeme několik možností, jak lze využít TEXovských databází v praxi, a to na výsledcích matematické soutěže.

Největším přínosem práce pro mě je hlubší pochopení činností systému TEX, bez jehož zvládnutí (alespoň na uživatelské úrovni) není vzdělání absolventa matematického oboru úplné – každý musí být schopen své myšlenky publikovat v přijatelné podobě (a proč ne v té nejkrásnější formě?). Dále doufám v to, že by moje práce mohla být užitečná lidem, kteří budou mít zájem seznámit se s TEXovskými databázemi, a to jako literatura k samostudiu.

1. TEX

Program TEX vytvořil Donald Ervin Knuth, profesor Stanfordovy Univerzity, expert v oboru výpočetní techniky. Slovy Donalda E. Knutha: je *určený pro tvorbu hezkých knih — a obzvlášť knih obsahujících spoustu matematiky.* TEX je volně šiřitelný „profesionální sazeč“, je nepostradatelným pomocníkem matematiků, fyziků a informatiků, jimiž je velice oblíben. TEX je nezávislý na operačním systému a na výstupním zařízení, tzn. pracuje se stejným výsledkem na různých výpočetních systémech a s libovolným operačním systémem, přitom je lhostejné, zda bude výsledek tisknut na laserové tiskárně nebo půjde přes osvitovou jednotku na film.

TEX byl naprogramován pro sazbu anglických textů, lze ho však přizpůsobit k sazbě v jakémkoliv jazyku. Do textů lze bez problémů vkládat akcenty, používat jiné než latinkové abecedy, sázet zprava doleva, jako je tomu například v arabštině, nebo sázet do sloupců, jako je tomu v japonštině. Je tak vhodný i k sazbě textů humanitně zaměřených, dává dokonalý výsledek. Písmo Computer Modern implicitně použité TEXem pro sazbu je rovněž dílo Donalda Ervina Knutha (je jím sázena tato práce), lze však použít v podstatě každé digitalizované písmo. Do TEXovských dokumentů lze rovněž vkládat postskriptové elementy (obrázky, písma, kód).

Na rozdíl od editorů typu Word (WYSIWYG — What You See Is What You Get — systém), je TEX „mark up“ systém. Autor přímo do svého textu vyznačuje pomocí „značek“ požadavky na to, co by se s textem při sazbě mělo stát. Značky mají formu zpětného lomítka následovaného posloupností znaků majících význam písmena a končících mezerou (případné další mezery jsou ignorovány) nebo jedním znakem, který není písmeno. Co je a co není písmeno lze definovat pomocí tzv. kategorie znaku. Značky jsou v podstatě optické (autor chce ovlivnit vzhled sazby) nebo logické (autor si jimi usnadňuje zápis textu). Značky lze programo-

vat, neboť TEX je též programovací jazyk. Textový soubor obsahující TEXovské značky se nazývá *dokument*.

Filozofie TEXu (definovaná Knuthem) vyžaduje, aby při svém spuštění program nejprve načetl *formát* — knihovnu základních značek, které autor v textu může použít. Samotný TEX obsahuje ve svém jádru jen asi 300 značek, tzv. primitivů, představujících jakousi bázi sazebních pokynů. S těmi by se nepracovalo dobře, jsou vhodné jen pro zkušené typografy programátory. Knuth proto naprogramoval první formát, tzv. *plain* a zakázal ho dále upravovat. Dokument napsaný v plainu musí dát i za 1 000 roků stejný výsledek. TEX spuštěný s formátem plain se nazývá PlainTEX.

PlainTEX je uživatelsky přítulný ovšem leda tak pro programátory. Pro běžného uživatele je stále dosti náročný. Roku 1985 sestavila skupina tvůrců kolem Leslieho Lamporta formát LATEX, který je k běžnému uživateli přítulnější. Tento formát není zakonzervován, ale je průběžně aktualizován. Dnešní LATEX se tak od toho původního značně liší, mnoho značek je doplněno či rozšířeno. Proto byl „původní“ LATEX přejmenován na LATEX 2.09, současný se nazývá LATEX 2_ε, budoucnost za horizontem má jméno LATEX 3. Pomocí formátu LATEX může i uživatel bez znalosti sazby nebo programování získat během pár hodin, dnů či týdnů kvalitní sazbu svého díla. Lze ovšem právem požadovat, aby se každý uživatel LATEXu se základními typografickými pravidly seznámil — každá typografická chyba v TEXu patřičně vynikne.

Aby nebyl TEX závislý na výstupním zařízení, je hlavním výstupem z programu soubor obsahující abstraktní popis vysázených stránek, navržený „velkým čarodějem“ Donaldem Ervinem Knuthem, tzv. *dvi* soubor (device independent). Tento soubor byl dříve zpracováván ovladačem příslušným danému výstupnímu zařízení (laserové tiskárny s rozlišením 300 dpi, laserové tiskárny s rozlišením 600 dpi, inkoustové tiskárny, osvitové jednotky 1 200 dpi atd.), autor získal vysázené dílo v materiální podobě. Musel ovšem při zpracování dvi souboru vědět, který ovladač použít. V současné době se prosadil popis vzhledu stránky společnosti Adobe — postskript. Proto se dvi soubor zpracovává ovladačem *dvips*, výsledkem

je postkriptový soubor, s nímž většina zařízení dokáže pracovat. A jestliže ne, pak existují volně šiřitelné substituty postskriptu, např. GhostScript s preprocesorem GhostView, psView aj. Vydavatelství pak většinou požadují pdf[1] soubor vygenerovaný z postskriptu. Nikoliv pdf soubor jako výsledek alternativního výstupu TEXu! Ten lze získat také pomocí GhostScriptu nebo prostřednictvím webového serveru na stránce `http://ps2pdf.com/convert.htm`. Nejlepším řešením je použití komerčního programu Acrobat Distiller společnosti Adobe, ten ovšem není volně šiřitelný.

Na trhu se objevuje velké množství komerčních firem, které občas nabízejí své produkty za nepřiměřené ceny. Navíc se tyto produkty musí také za nemalý finanční obnos aktualizovat. Lze říci, že se tyto komerční předražené produkty nemohou s TEXem srovnávat, stačí porovnat výslednou sazbu.

Instalace TEXu je na internetu k dispozici zdarma, stejně jako spousta návodů na řešení méně běžných situací a studijní literatury — veškerá potřebná dokumentace je ostatně součástí TEXu.

TEX vyžaduje ke své činnosti spoustu dalších pomocných programů, které podle potřeby volá, není to jediný program, ale rozsáhlý systém. Systém TEX je dnes šířen především prostřednictvím dvou distribucí, tzv. TEXLive nebo MikTEX, lišících se především způsobem instalace. Rozdíly v uspořádání adresářů jsou dány odlišným přístupem systému k práci s fonty. MikTEX zachovává původní přístup, písma různých velikostí (generovaných systémem ze zdrojových textů) mají stejný název, např. cmr10.pk, a jsou uložena v odlišných adresářích pojmenovaných podle rozlišení písma, např. DPI600. TEXLive odlišuje písma různých velikostí příponou s číslem značícím rozlišení, např. cmr10.600pk. MikTEXvyžaduje online přístup na internet, jeho komponenty se doinstalovávají podle aktuální potřeby uživatele, např. použité značky v dokumentu. TEXLive se obvykle instaluje rovněž z internetu (lze pořídit též instalační CD), uživatel při instalaci volí, které komponenty chce nainstalovat a není-li si jist, může zvolit úplnou (full) instalaci. Systém TEXLive lze kdykoliv aktualizovat pomocí programu „TeX Live

[1] pdf je rovněž formát firmy Adobe, je to „okrájený“ postskript bez programovacího jazyka a bez vložených fontů, původně určený pro internetovou výměnu textů.

Manager“. Pro českého uživatele je v současnosti v TEXLive lépe řešena podpora českého jazyka, MikTEX příliš spoléhá na vytvářenou multijazykovou podporu sazby prostřednictvím „babylonské věže“ (se všemi riziky) — knihovnu *babel*.

1.1. Možnosti a omezení TEXu

Jako každý počítačový program, tak i tento má spoustu výhod, ale bohužel i nevýhod. Záleží však na každém uživateli, zda využije možností, které mu program nabízí, či se jím nechá omezit.

1.1.1. Možnosti TEXu

Začněme možnostmi TEXu.

- Uživatel nemusí být profesionál v oblasti programování a typografické sazby, přesto je schopen vysázet text pomocí lehce srozumitelných příkazů.
- Sazba v nejvyšší kvalitě. Optimalizovaný řádkový zlom. Precizní, rychlé, bezchybné zpracování a vysázení je zaručeno.
- Pro přípravu zdrojového textu (dokumentu) lze použít jakýkoliv textový editor (nejlépe ASCII).
- Systém je stabilní — od roku 1986 se prakticky nezměnil.
- Je zajištěna snadná přenositelnost dokumentů, nezávislost na operačním systému či specifickém počítači, bezpečná je i archivace dokumentů.
- Typografie má za sebou několik století vývoje, pravidla jsou závislá na jazykové, resp. geografické, poloze. TEX tato pravidla může respektovat.
- Snadná sazba matematických vzorců – sazba je lineární, formule lze kopírovat a vkládat stejně jako prostý text.
- Snadné vytváření tabulek, vkládání poznámek pod čarou, psaní rovnic, vkládání obrázků.

- Automatizovaná tvorba obsahu, indexu, seznamu obrázků a tabulek.
- Možnost logických odkazů na číslované objekty a stránky (rovnice, seznam literatury apod.).
- K dispozici jsou balíčky maker umožňující práci splňující standard daného vydavatele.
- Díky CsTUGu je vytvořeno velmi dobré zázemí pro českou sazbu (formát csplain, cslatex, pdfcslatex).
- TEX je vyzdvihován hlavně díky sazbě matematiky, má však uplatnění také v humanitních vědách, díky kvalitě sazby. (Po vzoru Gutenberga také první Knuthem vysázená kniha byla Bible.)
- Kromě standardního výstupu (csplain, cslatex) do DVI souboru (abstraktní popis vzhledu stránky podle Knutha) a dnes obvyklého převodu DVI do postskriptu existuje alternativní výstup do PDF (pdfcsplain, pdfcslatex).
- Zkušeným uživatelům otevírá „nekonečné“ možnosti díky zabudovanému programovacímu jazyku.
- I přes všechny tyto možnosti je TEX zdarma.

1.1.2. Omezení TEXu

Existují také určitá omezení.

- Aby byl uživatel schopen vysázet text skutečně kvalitně, musí znát o systému TEX mnohem více, než jen značky TEXu. Musí dobře znát typografická pravidla, neboť každá (nejen) typografická chyba se výrazně projeví na vzhledu sazby. Uživatel se tedy musí neustále vzdělávat. Zvládnutí TEXu na skutečně dobré úrovni navíc vyžaduje učit se ho „cyklycky“, TEX se nelze naučit „lineárně“ jediným prostudováním „uživatelské příručky“.

- Systém TEX není WYSIWYG, což může být zejména začátečníkům nepříjemné. Dokument je třeba nejprve TEXem zpracovat, teprve poté si je možné prohlédnout výsledek na monitoru, popřípadě si ho vytisknout.

- TEX se sice dokáže přizpůsobit národnímu prostředí, musí mu to ale být „řečeno“. Nedokáže však opravit chyby v textu.

- Zvládne jen velmi jednoduché výpočty, není totiž určen k matematickým výpočtům, ale k sazbě (nejen) matematiky.

- Standardní výstup vyžaduje odlišný formát vkládaných obrázků (.ps, .eps), než alternativní pdf výstup (.pdf, .png, .jpg) – neexistuje společný formát. Je nutné používat konverzní programy, (např. volně šiřitelný GIMP, ImageMagick apod.).

- Instalace programu zabírá poměrně velký prostor na disku. Množství souborů v distribuci TEXu způsobuje jejich nepřehlednost, a to i přes promyšlený systém adresářů.

2. Databáze

Databáze můžeme chápat jako soubor informací o určité skupině objektů (např. knih v knihovně, pacientů v nemocnici apod), který byl vytvořen pomocí počítače. S těmito soubory můžeme manipulovat — číst, upravovat a ukládat. Předchůdcem takovéto databáze byly obyčejné kartotéky, uspořádané papírové listy, obsahující informace, seřazené podle určitých zásad.

K vytvoření databáze je potřeba databázového programu. Jestliže tento program máme, je dalším krokem vytvoření uspořádaného souboru s informacemi podle našeho určitého záměru. Systém řízení bází dat, jak sám název napovídá, řídí naše data a umožňuje práci s databází. Výslednou podobu databáze následně uložíme do paměti počítače. Máme-li databázi vytvořenou a uloženou, můžeme s ní začít pracovat. S databází pracujeme prostřednictvím dotazovacích jazyků. Ty nám pomáhají s výběrem požadovaných dat z databáze (např. knih určitého autora, pacienta dané diagnózy apod). Každá databáze má svého správce — údržbáře databáze — který databázi rozumí a má pravomoc ji upravovat a nakládat s ní.

3. Databáze v TEXu

Zmiňovali jsme se o tom, že k vytvoření databáze je potřeba databázového programu. To ale není docela pravda. Myšlenku, že databázi lze vytvořit bez speciálního databázového programu použil Petr Olšák, viz [4], propagátor TEXu a významný představitel Československého sdružení uživatelů TEXu ($\mathcal{CS}$Tug). Soubor informací o určité skupině objektů (databázi) lze vytvořit i textovým editorem a program TEX dokáže text zpracovávat podle značkami definovaných činností. S textovou databází se pracuje velice snadno, je přehledná a je ji možné snadno zveřejnit. Ne každý je ale uživatelem TEXu, většina lidí tento program nemá ve svém počítači vůbec nainstalován, tudíž s textovou databázi nemohou pracovat, i když prohlédnout si ji mohou. Zkušený uživatel TEXu však může s takovouto textovou databází pohodlně pracovat.

3.1. Ukázka jednoduché databáze

V této kapitole uvedeme jednoduchý příklad, na kterém ukážeme vše potřebné pro přípravu TEXovské databáze a práci s ní. Textovou databází je např. následující dokument:

```
\newcount\globnum
\def\p#1|#2|#3| {\advance\globnum by 1
\begingroup
\jmena #1|\adresa #2|\info #3|%
\expandafter\test\expandafter\print\endtest
\endgroup}

\def\jmena #1 #2:#3#4#5#6#7#8/#9|{%
\def\prijmeni{#1}%
\def\jmeno{#2}%
\def\rc{#3#4#5#6#7#8/#9\unskip}
\def\den{#7#8\unskip}\def\mesic{#5#6\unskip}\def\rok{#3#4\unskip}}

\def\adresa #1:#2:#3:#4|{%
\def\ulice{#1\unskip}\def\mesto{#2}%
\def\cp{#3\unskip}\def\psc{#4\unskip}}%

\def\info #1.#2.#3|{%
\def\oddne{#1\unskip}%
\def\odmesice{#2}%
\def\odroku{#3\unskip}}%

\let\test=\relax \let\endtest=\relax
\def\reserse#1#2{\def\test{#1}\def\endtest{#2}}
\reserse{\relax}{\relax}
\def\empty{ \unskip}

\def\print {%
\vbox{ \noindent\llap{\the\globnum. }
 \makebox[0.35\hsize][l]{{\bfseries \prijmeni\ \jmeno}}
 \makebox[0.7\hsize][c]{RČ: \rc}  \\
Bydliště: \ulice{} \cp , \mesto  \\
 \makebox[0.25\hsize][l]{PSČ: \psc} \\
Účast od \oddne.   \odmesice.    19\odroku
\par\vspace{3.5ex} } }
```

```
\globnum = 0
\subsection*{Členové}

\p Novák Jiří:760512/2218
| Českobrodská:Praha:1250:110 00
| 28.8.90|

\p Svoboda Pavel:660620/5704
| Olomoucká:Brno:987:602 00
| 30.6.86|

\p Dvořák Miroslav:650307/3208
| Průmyslová:Olomouc:1498:779 00
| 3.6.95|

\p Černý Jaroslav:700409/4217
| Ocelkova:Šternberk:138:785 01
| 5.11.93|

\p Procházka Josef:530113/1743
| Liberecká:Brno:4380:602 00
| 27.5.76|

\p Veselý František:731224/0841
| Hřenská:Olomouc:9067:779 00
| 1.9.93|

\p Horák Jan:681104/2654
| Čakovická:Litovel:9012:784 01
| 8.3.97|

\p Hájek Petr:700912/4160
| Prosecká:Šumperk:569:787 01
| 24.1.98|
```

A jaký dostaneme výsledek, zpracujeme-li tento dokument LaTeXem?

Členové

1. **Novák Jiří** RČ: 760512/2218
   Bydliště: Českobrodská 1250, Praha
   PSČ: 110 00
   Účast od 28. 8. 1990

2. **Svoboda Pavel** RČ: 660620/5704
   Bydliště: Olomoucká 987, Brno
   PSČ: 602 00
   Účast od 30. 6. 1986

3. **Dvořák Miroslav** RČ: 650307/3208
   Bydliště: Průmyslová 1498, Olomouc
   PSČ: 779 00
   Účast od 3. 6. 1995

4. **Černý Jaroslav** RČ: 700409/4217
   Bydliště: Ocelkova 138, Šternberk
   PSČ: 785 01
   Účast od 5. 11. 1993

5. **Procházka Josef** RČ: 530113/1743
   Bydliště: Liberecká 4380, Brno
   PSČ: 602 00
   Účast od 27. 5. 1976

6. **Veselý František** RČ: 731224/0841
   Bydliště: Hřenská 9067, Olomouc
   PSČ: 779 00
   Účast od 1. 9. 1993

7. **Horák Jan** RČ: 681104/2654
   Bydliště: Čakovická 9012, Litovel
   PSČ: 784 01
   Účast od 8. 3. 1997

8. **Hájek Petr** RČ: 700912/4160
   Bydliště: Prosecká 569, Šumperk
   PSČ: 787 01
   Účast od 24. 1. 1998

Vidíme, že databáze má opravdu podobu textového souboru a že výstupem je rovněž text. Textové databáze umožňují zpracovávat data o čemkoli, ať už jsou předmětem našeho zájmu lidé, knihy, zvířata, rostliny, města, atd. Jsou vhodné tam, kde chceme pěkný tiskový výstup, nejlépe v kvalitě knižní sazby.

Nyní na této jednoduché databázi vysvětlíme vše potřebné. Nejprve si všimněme, že *databáze je složená ze dvou logických částí. První část se týká definování struktury databáze a druhá část jsou potom samotná data*, v našem případě data o členech nějakého sdružení.

Ještě dříve, než se pustíme do tvoření samotné TEXovské databáze, bychom se měli zamyslet nad strukturou databáze a nad tím, jak bude vypadat výstup z databáze (vysazený text).

TEXovské databáze obvykle nemají složitou strukturu, přesto musíme dbát na to, abychom se v její struktuře neztratili. Proto bychom si budoucí strukturu databáze měli velmi dobře promyslet — třeba i na papír načrtnout její strukturu. Promyšlení, uspořádání a načrtnutí struktury databáze je výhodné například k tomu, aby databáze byla přehledná a srozumitelná. Je vhodnou pomůckou zejména při vytváření první části databáze, o které již byla řeč. Nejdříve totiž tvoříme jakousi formu, do které až nakonec vkládáme data.

Při budování TEXovské databáze využíváme možnost, kterou nabízí definování značky s parametrem v plainTEXu, a tou je použití oddělovacích znaků parametru, LATEXovské definice toto neumožňují. Při volání takovéto značky v TEXovském

programu jsou všechny znaky následující za značkou až po tento oddělovací znak načteny jako její parametr a může s nimi být dále pracováno. Jednotlivé členy databáze musíme zadávat právě v takto zvolené formě, s příslušnými oddělovacími znaky. Proto je velmi důležité si na počátku přípravy databáze formu vstupu dobře promyslet.

Značka `\p`, neboli makro či příkaz, udává hrubou strukturu textové databáze

```
\def\p#1|#2|#3| {\advance\globnum by 1
\begingroup
\jmena #1|\adresa #2|\info #3|%
\expandafter\test\expandafter\print\endtest
\endgroup}
```

Má tři parametry, každý ukončený námi zvoleným znakem `|`.

```
\def\p#1|#2|#3|
```

Místo svislých lomítek lze použít i jiné znaky jako je např. dvojtečka, tečka apod., či dokonce značky, které však při takovém použití ztrácejí svoji funkci, pouze oddělují parametry.

První parametr (`#1`) se týká jména člena, druhý (`#2`) adresy a třetí (`#3`) obsahuje nějaké další informace. Značky `\begingroup` a `\endgroup` mají význam programátorských závorek, vyznačují blok — všechny značky definované v tomto bloku platí jenom v něm (není-li řečeno jinak), volání všech maker je platné pouze v rámci bloku.

Znak `%` uvozuje poznámku, od tohoto znaku až do konce řádku jsou všechny znaky ignorovány, nehrozí tak, že by nějaký „neviditelný“ znak způsobil chybu při zpracování databáze. Neviditelné jsou např. mezery, tabelátory a tzv. invalidní znaky, definované v TEXu.

Čítač `globnum`, deklarovaný příkazem `\newcount\globnum`, jsme se rozhodli použít k očíslování jednotlivých členů sdružení v databázi (členů databáze). Čítače jsou nezáporné celočíselné proměnné, deklarovaný čítač má implicitní hodnotu 0. Výraz `\advance\globnum by 1` zvětšuje hodnotu čítače o 1, každý člen databáze bude mít své číslo.

K dokončení vysvětlení funkce první značky \p zbývá ještě jeden řádek:

```
\expandafter\test\expandafter\print\endtest
```

Tímto řádkem se dotazujeme na členy databáze. Dotaz je formulován ve značce \test. Protože k dotazům se používají podmíněné příkazy \if, \ifx, \ifnum aj., je třeba je ukončit odpovídající značkou, většinou \fi, proto značka \endtest. Tato značka ovšem může být i prázdná (požadujeme-li výstup všech členů), proto ji nelze definovat rovnou \fi.

Značka \expandafter funguje právě tak, jak se očekává od jejího jména. Expand procesor TEXu nahradí při každém cyklu zpracování dokumentu každou značku obsahem těla její definice. A cyklů vykoná právě tolik, aby v expandovaném dokumentu již žádná značka (mimo primitivů) nebyla. Místo \p se tedy po prvním průchodu expandprocesorem objeví:

```
\advance\globnum by 1
\begingroup
\jmena #1|\adresa #2|\info #3|%
\expandafter\test\expandafter\print\endtest
\endgroup
```

Značka \jmena #1 (rovněž tak \adresa #2 a \info #3) ještě neexpandovala, není tedy rozebrán její parametr na jednotlivá data, a tato data v tomto okamžiku nejsou k dispozici. Expanze značky \test by tak způsobila chybu. Značka \expandafter zamezí expanzi následující značky, expandprocesor ji přeskočí a bude muset vykonat další cyklus, a to již budou data od značek \jmena, \adresa a \info k dispozici. V opačném případě bychom museli počet značek \expandafter ztrojnásobit.

Nyní se podívejme hlouběji do naší textové databáze. Základní koncept databáze je zakotvený ve značce \p. Její tři parametry obsahují po řadě údaje o jméně, adrese a další informace (dejme tomu o počátku evidence v databázi). Tyto tři parametry jsou načteny značkami \jmena #1, \adresa #2 a \info #3, které je rozeberou na jednotlivá data. Lze namítnout, že tuto činnost mohla rovnou vykonat značka \p. Jistěže mohla, ale pouze v případě, že by počet parametrů nebyl

větší než 9. Námi představený koncept umožňuje použít v databázi 9×9 jednotlivých údajů, což většinou postačuje. V opačném případě bychom museli do naší databáze přidat další úroveň parametrů.

Název každé značky volíme pokud možno tak, aby vypovídal o datech jí zpracovávaných. Lépe se pak v databázi orientujeme a lépe se v ní orientují i další lidé, kteří s databází pracují. V rozsáhlejších textových databázích to uvítáme.

V naší ukázkové databázi pohltila první parametr značka `jmena`. Druhý parametr pohltila značka `adresa`, jejíž název mluví sám za sebe, a třetí parametr, pohlcený značkou `info` bude prozrazovat nějaké dodatečné informace. Definice těchto značek je analogická definici značky `\p`, značka je definována s parametry ukončenými oddělovači. V těle definic však již nejsou parametry předávány značkám k dalšímu zpracování, ale jsou rovnou vhodné značce přiřazeny, např. `\def\prijmeni{#1}`. Na jednotlivá data se tak budeme obracet jejich jménem — data jsme pojmenovali.

Nyní se opět vrátíme k naší databázi. Struktura parametru `#1` značky `\p` je `#1 #2:#3#4#5#6#7#8/#9`, je tedy složena z dalších devíti parametrů. Data načtená tímto parametrem mají tvar např. `Svoboda Pavel:820620/5704`. Značka, rozebírající tento parametr, je:

```
\def\jmena #1 #2:#3#4#5#6#7#8/#9|{%
 \def\prijmeni{#1}%
 \def\jmeno{#2}%
 \def\rc{#3#4#5#6#7#8/#9\unskip}
 \def\den{#7#8\unskip}
 \def\mesic{#5#6\unskip}
 \def\rok{#3#4\unskip}}
```

Parametr značky `jmena` je ukončen svislým lomítkem, přesně tak, jak jsme si naplánovali. Nejdříve zadáváme příjmení. To, že zadáváme příjmení poznáme z druhého řádku, kde je nadefinováno, že příjmení bude odpovídat parametru číslo jedna. Příjmení končí mezerou a následuje druhý parametr. Druhý parametr je načten ve třetím řádku a skrývá se pod ním jméno. Jméno je zakončeno dvojtečkou. Za dvojtečkou je vypsáno rodné číslo. Rodné číslo je v naší databázi

zapsáno poněkud složitě. Pro naše účely by pro ně plně postačil jediný parametr namísto sedmi. To by zápis vypadal takto.

```
\def\jmena #1 #2:#3|{%
```

Rodné číslo by se skrývalo v třetím parametru. Samozřejmě bychom museli upravit i definici značky `\rc` ve čtvrtém řádku, kde by se objevil pouze parametr `#3`. Tento zápis by vyhovoval a rodné číslo není problém vypsat. Kdybychom si s parametrem *rodné číslo* chtěli ještě pohrát, mohli bychom ho rozdělit například na dvě části – mezi dva parametry. Tyto parametry bychom si mohli oddělit jakoukoliv značkou – my jsme zvolili lomítko. Odpovídající definice pak vypadá takto:

```
\def\jmena #1 #2:#3/#4|{%
```

Z tohoto zápisu plyne, že do třetího parametru bychom vkládali první šestičíslí rodného čísla a čtvrtý parametr by obsahoval poslední čtyři číslice rodného čísla. Ještě bychom museli upravit čtvrtý řádek, který by vypadal takto:

```
\def\rc{#3/#4\unskip}
```

V databázi, kde jsme použili pro rodné číslo jeden či dva parametry, nemůžeme přímo zjistit den, měsíc, rok narození nebo pohlaví jeho držitele — třeba kvůli oslovení „pane“ nebo „paní“. Museli bychom makrem „zhltnutý“ řetězec, tzv. token, dále pracně rozebírat. Způsob zpracování rodného čísla, uvedený v naší databázi, nám dává k dispozici den, měsíc, rok narození i pohlaví člena databáze okamžitě. V naší ukázce značka `\prijmeni` značí *Svoboda*, značka `\jmeno` znamená *Pavel*, . . . , `\rok` znamená *82*.

Značka `adresa`, přebírající druhý parametr, je definována obdobně jako `jmena` a funguje na stejném principu.

```
\def\adresa #1:#2:#3:#4|{%
 \def\ulice{#1\unskip}%
 \def\mesto{#2}%
 \def\cp{#3\unskip}
 \def\psc{#4\unskip}}%
```

Pod prvním parametrem se skrývá název ulice. Je ukončen dvojtečkou. Následuje druhý parametr, který udává město, opět je ukončen dvojtečkou, stejně jako třetí parametr, udávající číslo popisné. Vše zakončuje čtvrtý parametr, obsahující poštovní směrovací číslo.

Poslední značka `info` převezme informaci o tom, od jakého data se lidé stali členy naší databáze. Definice této značky je tato:

```
\def\info #1.#2.#3|{%
\def\oddne{#1\unskip}%
\def\odmesice{#2}%
\def\odroku{#3\unskip}}%
```

Parametry této značky jsou oddělené tečkami. Kvůli odlišení od data narození z rodného čísla je název značky modifikován předponou „od“.

Podíváme-li se na samotné zadávání členů databáze, jsou data o každém jejím členu uložena v parametrech značky `\p`.

```
\p Novák Jiří:760512/2218
| Českobrodská:Praha:1250:110 00
| 28.8.90|
```

Hlavní parametry, zpracovávané značkami `\jmena #1`, `\adresa #2` a `\info #3` jsou vždy ukončeny svislým lomítkem. Také jejich vnitřní struktura je zapsána právě tak, jak bylo definováno.

Zbývá rozebrat ještě jednu část definic, a to:

```
\let\test=\relax \let\endtest=\relax
\def\reserse#1#2{\def\test{#1}\def\endtest{#2}}
\reserse{\relax}{\relax}
\def\empty{ \unskip}
```

Zde je definováno, co dělají značky `\test` a `\endtest`. Stručně řečeno, na počátku nic — relaxují. Pro jejich snadné nastavení je pak definována značka `reserse` se dvěma „klasickými“ parametry — bez použití oddělovačů. Parametry tedy musí být uvedeny ve složených závorkách. Značka je inicializována na hodnotu `\relax`. Značka `\empty` je synonymem pro mezeru následovanou značkou `\unskip` a je využívána pro testování toho, zda je zpracovávaný parametr prázdný či nikoliv.

Tím je téměř vše potřebné nadefinováno. Zbývá definovat vzhled výstupu `\print` z databáze:

```
\def\print {%
\vbox{ \noindent\llap{\the\globnum. }
 \makebox[0.35\hsize][l]{{ \bfseries \prijmeni\ \jmeno}}
 \makebox[0.7\hsize][c]{RČ: \rc}  \\
Bydliště: \ulice{} \cp , \mesto  \\
 \makebox[0.25\hsize][l]{PSČ: \psc} \\
Účast od \oddne.   \odmesice.    19\odroku
\par\vspace{3.5ex} }%endvbox
}%endprint
```

Značka obsahuje standardní příkazy LaTeXu, snad se jen zmíníme o tom, že primitiv `\the` vysází obsah čítače `\globnum` a `\llap` je box nulové šířky vysazený vlevo od aktuálního bodu sazby.

Může se stát, že nebudeme znát o každém členovi databáze všechny informace. V tom případě dodržíme nadefinovanou formu zápisu parametrů, jen odpovídající parametr necháme prázdný nebo do něho vložíme mezeru. Takový člen databáze bude bez jakéhokoliv problému vysázen:

```
\p Novák  :760512/2218
| Českobrodská:Praha:1250:110 00
| 28.8.90|

\p Svoboda Pavel:660620/5704
| Olomoucká: :987:602 00
| 30.6.86|
```

U prvního člena databáze jsme úmyslně vynechali jeho jméno. Za příjmením jsme ovšem nezapomněli na mezeru — tím zachováme strukturu databáze (formálně je jméno a příjmení odděleno mezerou). Místo jména jsme také vložili mezeru, ta zde ale být nemusí. Parametr je ukončen dvojtečkou. Druhému členovi chybí údaj *mesto*, je nahrazen mezerou, následuje dvojtečka (parametr je obehnán dvojtečkami). Výsledek je pak tento:

Členové

1. **Novák** RČ: 760512/2218

   Bydliště: Českobrodská 1250, Praha

   PSČ: 110 00

   Účast od 28. 8. 1990

2. **Svoboda Pavel** RČ: 660620/5704

   Bydliště: Olomoucká 987,

   PSČ: 602 00

   Účast od 30. 6. 1986

Čárku za ulicí *Olomoucká 987* je možné nevysazovat, stačí otestovat, zda je parametr prázdný (do parametru vložíme `\unskip`), a zařídit se podle toho — využít podmíněné příkazy (větvení):

```
def\print {%
\vbox{ \noindent\llap{\the\globnum. }
 \makebox[0.35\hsize][l]{{\bfseries \prijmeni\ \jmeno}}
 \makebox[0.7\hsize][c]{RČ: \rc}  \\
Bydliště: \ulice{} \cp \ifx\mesto\empty\else,\fi \mesto  \\
 \makebox[0.25\hsize][l]{PSČ: \psc} \\
Účast od \oddne.   \odmesice.    19\odroku
\par\vspace{3.5ex} } }

\p Svoboda Pavel:660620/5704
| Olomoucká: \unskip:987:602 00
| 30.6.86|
```

1. **Svoboda Pavel** RČ: 660620/5704

   Bydliště: Olomoucká 987

   PSČ: 602 00

   Účast od 30. 6. 1986

3.2. Podmíněné příkazy

Podstatnou vlastností databáze je schopnost „odpovídat na dotazy“ sestavou obsahující členy, na které byl učiněn dotaz. Vypsat např. ty, které spojuje místo bydliště, rok narození, dodatečná informace. Ukážeme na několika příkladech, jak tuto situaci řeší naše textová databáze.

K zadání dotazu slouží značka `\test`, která je i se svým ukončením `\endtest` definována pomocí značky `\reserse`. Obsahem `\testu` je některý z podmíněných příkazů, jež jsou součástí programu TEX a jsou použitelné k dotazu na členy databáze. Jsou to značky:

1. `\ifx <token1><token2>` — jediný podmíněný příkaz, kde se `<token1>` i `<token2>` pro zpracování podmínky berou v neexpandovaném tvaru. Podmínka je pravdivá, jestliže (a) oba tokeny nejsou makra a oba reprezentují znak stejného ASCII kódu i kategorie současně (b) oba tokeny jsou značky naprosto stejného významu, tj. mají stejnou masku parametru a jejich těla obsahují stejnou posloupnost tokenů a makra jsou definována se stejným prefixem, např. `\long`. V tomto případě se vykoná `<true text>`, jinak `<false text>`.

Syntaxe příkazu je

```
\ifx <token1><token2> <true text> [ \else <false text> ] \fi
```

Např., viz [3, str. 377]:

```
\def\a#1.{a} \def\b#1:{a} \long\def\c#1:{a}
\def\d{a} \edef\e{a} \let\f=\kern \def\g{\kern}
\ifx\kern\f Ano, to je pravda.\fi
\ifx\kern\g To není pravda.\fi
\ifx\d\e To je pravda.\fi
% ale ostatní \a, \b, \c, \d se vzájemne z pohledu \ifx liší
\ifx\kern\hbox To tedy pravda není.\fi
\ifx AA To je fakt.\fi
```

2. `\if<token1><token2>` — Je-li ASCII hodnota `<token1>` rovna ASCII hodnote `<token2>`, bude test pravdivý, v opačném případě bude test nepravdivý. Podmínka `<token1><token2>` se sestaví až po úplné expanzi. Řídící sekvence,

které přijaly význam nějakého znaku (nebo sekvence) pomocí `\let` se považují za shodné s tímto znakem (nebo sekvencí). Ostatní (neexpandovatelné) řídící sekvence mají hypotetickou ASCII hodnotu 256. Kategorie `<token1>` a `<token2>` se při vyhodnocení podmínky ignorují.

Např., viz [3, str. 375]:

```
\if \kern\relax Ano to je pravda, protože 256=256.\fi
\let\hvezda=*
\if *\hvezda Ano, to je také pravda.\fi
\def\c{Aa}
\if \c Neplatí, protože ASCII "A" není rovno ASCII "a".\fi
```

Syntaxe příkazu je

```
\if <token1><token2> <true text> [ \else <false text> ] \fi
```

3. `\ifnum<number1><relation><number2>` — porovnává hodnotu dvou čítačů `<number1>` a `<number2>`, `<relation>` musí být $<$, $=$, $>$. Zbývající relace lze implementovat pomocí negace (činnost za `\else`).

Syntaxe příkazu je

```
\ifnum <number><relation><number2> <true text>
  [ \else <false text> ] \fi
```

4. `\ifcase<number><text for case 0>\or<text for case 1>\or...`
`\or<text for case n>\else<text for all other cases>\fi`

(vícenásobné větvení) — větvení podle hodnoty `<number>` na $n+1$ větví. `<case i>` jsou tokeny, které se provedou v případě, že `<number>=i`. Mezi `<case 0>` až `<case n>` je použito n separátorů `\or`. Není-li `<number>` v intervalu 0 až n, provede se případně `others`.

5. Protože TEX umožňuje definovat logické proměnné, lze přímo testovat jejich hodnotu. Jméno logické proměnné tvoří posloupnost písmen povinně začínající znaky `\if`, např. `\ifstudent`, `\ifclen` apod. Jejich hodnoty *true*, resp. *false*, se nastaví voláním značky vytvořené ze zvolené posloupnosti písmen následované logickou hodnotou, např. `\studenttrue`, `\studentfalse` apod. Test pak má syntax `\ifstudent <text for case true> \else <text for case false> \fi`.

Obvykle se k výběru dat z databázeru užívá test 1, 3, 4 nebo 5. V případě 1 se porovnávají řídící sekvence, řídící znaky nebo přímo písmena. Např. definujeme `\def\ol{Olomouc}`, a pak porovnáváme obsah značky `\mesto` příkazem `\ifx\ol\mesto`. V případech 3 a 4 můžeme místo řídících sekvencí:

```
\def\pet{95}
\reserse{\ifx\pet\odroku}{\fi}
```

porovnávat čítače (celá čísla):

```
\newcount\ctyri \ctyri=94
\reserse{\ifnum\ctyri>\odroku}{\fi}
```

Nyní ukážeme použití podmíněných příkazů na naší databázi. Vyberme nejprve ty členy databáze, kteří mají jako bydliště uvedenu Olomouc. K tomu využujeme podmíněný příkaz `\ifx`. Postup bude následující. Nadefinujeme si značku *ol* s tělem Olomouc: `\def\ol{Olomouc}`. Tuto definici umístíme na začátek zdrojového kódu. Nyní nastavíme test: `\reserse{\ifx\ol\mesto}{\fi}`. Tento příkaz umístíme před definici tisku, význam značky `\test` musí `\print` znát. Takto nastavená rešerše vypíše z databáze pouze ty členy, u kterých je v parametru `\mesto` zadáno *Olomouc*.

Členové

3. **Dvořák Miroslav** RČ: 650307/3208
   Bydliště: Průmyslová 1498, Olomouc
   PSČ: 779 00
   Účast od 3. 6. 1995

6. **Veselý František** RČ: 731224/0841
   Bydliště: Hřenská 9067, Olomouc
   PSČ: 779 00
   Účast od 1. 9. 1993

Dostali jsme opravdu pouze ty členy, které jsme požadovali. Můžeme si všimnout, že vytřídění členové databáze mají stále svoje pořadové číslo. Nebude problémem je najít ani v rozsáhlých databázích.

Dalším příkladem je hledání osob se stejným křestním jménem. Postup je analogický. Nadefinujeme značku `\pa` znamenající Pavel a nastavíme test:

```
\def\pa{Pavel}
\reserse{\ifx\pa\jmeno}{\fi}
```

Značky samozřejmě umístíme na odpovídající místo.

Členové

2. **Svoboda Pavel** RČ: 660620/5704
   Bydliště: Olomoucká 987, Brno
   PSČ: 602 00
   Účast od 30. 6. 1986

Poslední příklad se bude týkat roku, kdy člověk vstoupil do databáze. Rešerše probíhají analogicky. Definujme značku `pet` vyjadřující rok vstupu do databáze — v našem případě to bude rok 1995:

```
\def\pet{95}
\reserse{\ifx\pet\odroku}{\fi}
```

Členové

3. **Dvořák Miroslav** RČ: 650307/3208
   Bydliště: Průmyslová 1498, Olomouc
   PSČ: 779 00
   Účast od 3. 6. 1995

Tyto jednoduché rešerše jsou velmi snadné. Lze ale samozřejmě sestavit i kombinace dotazů. Olomoučan, který byl členem databáze před rokem 1994, je zjištěn testem

```
\newcount\ctyri \ctyri=94
\def\ol{Olomouc}
\reserse{\ifnum\ctyri>\odroku\ifx\ol\mesto}{\fi\fi}
```

umístěným samozřejmě před definici `\print`. Tentokrát jsme dotaz na počátek členství řešili pomocí čítače.

Členové

6. **Veselý František** RČ: 731224/0841

   Bydliště: Hřenská 9067, Olomouc

   PSČ: 779 00

   Účast od 1. 9. 1993

4. Zpracování výsledků soutěže

Program TEX nedisponuje programovacími možnostmi k řešení matematických algoritmů, nevyrovná se tabulkovým kalkulátorům – nebyl napsán pro tyto účely. Jeho předností je možnost získat tiskový výstup nejvyšší možné kvality. Umožňuje také, jak bylo ukázáno v předešlé části, naprogramovat textovou databázi. Máme tak k dispozici silný nástroj pro automatizovaný tisk výsledků soutěže, diplomů pro účastníky aj. Výpočty, seřazení soutěžících, přidělení odměn (medailí a čestných uznání) přenecháme tabulkovému kalkulátoru, jehož výstup bude sloužit jako podklad pro zpracování TEXem. Vhodným výstupem pro další zpracování TEXem je typ souboru CSV – Excel nabízí uložit jako CSV (oddělený středníkem) (*.csv). Tento soubor pak můžeme vhodným editorem upravit, např. odstranit nepotřebná pole, zaměnit oddělovače, doplnit název TEXovské značky.

4.1. Tisk diplomů

Nechť má naše soutěž dvě části — soutěž jednotlivců a soutěž družstev (to je typické např. pro matematické či fyzikální olympiády). Ukážeme si, jak tisknout diplomy pro jednotlivce a diplomy pro tým. Použijeme data ze Středoevropské matematické olympiády z roku 2008, konané v Olomouci.

4.1.1. Tisk diplomů soutěže jednotlivců

Data soutěžících připravená a zpracovávaná v tabulkovému kalkulátoru (Excel) uložíme jako soubor CSV, např. `jednotlivci.csv`:

```
GER1;Arnold;Bertram;8;8;8;8;32;1;Gold;;;;;
HUN1;Éles;András;8;8;8;8;32;1;Gold;;;;;
HUN3;Nagy;Dániel;8;8;8;8;32;1;Gold;;;;;
POL5;Oćwieja;Jakub;8;8;8;8;32;1;Gold;;;;;
HUN5;Szűcs;Gergely;8;8;8;8;32;1;Gold;;;;;
POL1;Drobiński;Patryk;8;8;7;8;31;6;Silver;;;;;
POL4;Malinowski;Daniel;8;8;7;8;31;6;Silver;;;;;
POL3;Konaszyński;Karol;8;8;7;7;30;8;Silver;;;;;
GER5;Reinhold;Jens;8;4;8;8;28;9;Silver;;;;;
HUN2;Fonyó;Dávid;8;7;4;8;27;10;Silver;;;;;
```

GER2;Buchholz;Simon;8;8;0;8;24;11;Silver;;;;;
CZE1;Klaška;David;8;8;0;8;24;11;Silver;;;;;
POL2;Kociumaka;Tomasz;8;8;0;8;24;11;Silver;;;;;
GER3;Merker;Martin;8;8;0;8;24;11;Silver;;;;;
HUN4;Szőke;Nóra;8;8;0;8;24;11;Silver;;;;;
HUN6;Varga;László;8;8;0;8;24;11;Silver;;;;;
CRO2;Božić;Ivo;8;7;0;8;23;17;Bronze;;;;;
SVK3;Csiba;Peter;8;7;0;8;23;17;Bronze;;;;;
SVK4;Hagara;Michal;8;7;0;8;23;17;Bronze;;;;;
SUI6;Su;Pascal;8;8;0;7;23;17;Bronze;;;;;
SUI3;Dujmović;Hrvoje;8;8;0;6;22;21;Bronze;;;;;
AUT3;Hafner;Johannes;8;6;0;8;22;21;Bronze;;;;;
POL6;Orlef;Damian;8;6;0;8;22;21;Bronze;;;;;
CZE4;Pavlík;Tomáš;3;1;8;7;19;24;Bronze;;;;;
CRO4;Ljulj;Matko;1;8;0;8;17;25;Bronze;;;;;
AUT4;Müllner;Clemens;7;4;0;6;17;25;Bronze;;;;;
AUT5;Pfannerer;Stephan;8;1;0;8;17;25;Bronze;;;;;
SUI5;Pohle;Clemens;8;0;0;8;16;28;Bronze;;;;;
GER4;Recknagel;Judit;0;8;0;8;16;28;Bronze;;;;;
CRO6;Valentić;Grgur;0;8;0;8;16;28;Bronze;;;;;
SVK1;Bačo;Ladislav;8;0;0;7;15;31;HM;;;;;
SVK2;Bachratý;Martin;8;2;0;5;15;31;HM;;;
SVK6;Szaniszlo;Tomáš;8;7;0;0;15;31;HM;;;
SUI2;Cieslewski;Titus;6;8;0;0;14;34;HM;;;
AUT2;Fuchs;Adrian;5;1;0;8;14;34;HM;;;
SVK5;Konečný;Jakub;5;8;0;1;14;34;HM;;;
CRO3;Cicvarić;Borna;5;0;8;0;13;37;HM;;;
CZE2;Marek;Jiří;5;8;0;0;13;37;HM;;;
AUT1;Dräxler;Felix;4;7;0;1;12;39;;;;
SUI1;Bachmann;Jürg;3;8;0;0;11;40;HM;;;
GER6;Shevchenko;Radomyra;2;0;0;8;10;41;HM;;;
SLO4;Stepišnik Perdih;Tomaž;5;4;0;1;10;41;;;;
SLO3;Payrits;Matjaž;0;0;0;8;8;43;HM;;;
CZE6;Vaňhara;Jan;0;7;0;0;7;44;;;;
SLO2;Kozarski;Filip;5;0;0;0;5;45;;;;
CZE3;Nguyen;Van Nhan;4;1;0;0;5;45;;;;
CZE5;Šormová;Hana;5;0;0;0;5;45;;;;
AUT6;Roitner;Valerie;0;4;0;0;4;48;;;;;
SUI4;Lagger;Cyril;1;0;1;1;3;49;;;;;
SLO1;Breznik;Eva;0;1;0;1;2;50;;;;;
CRO1;Antoliš;Ivana;0;0;0;0;0;51;;;;;
CRO5;Telarović;Irma;0;0;0;0;0;51;;;;;

Překopírujeme ho do LaTeXovského dokumentu, podle potřeby upravíme a přidáme potřebná makra. V tomto případě vypadá LaTeXovský dokument takto:

```
\documentclass[a4paper,12pt]{article}
\usepackage[cp1250]{inputenc}
\usepackage[dvips]{graphics}
\usepackage{czech}

\tabcolsep0pt        %v tabulce je mezera mezi sloupci 0pt
\def\thepage{}       %nebudou se tisknout čísla stran
\voffset-1in         %posun nahoru o 1 palec
\hoffset-12mm        %posun doleva o 12 mm
\textwidth16cm
\textheight9.8in

\begin{document}

\newcount\typec      %typ ocenění

\def\p#1;#2;#3;#4{   %nadefinování struktury parametrů
\def\name{#2}
\def\surname{#3}
\typec=#4
\input telo.tex \clearpage
}

\p AUT;Felix;DRÄXLER;5
\p AUT;Adrian;FUCHS;4
\p AUT;Johannes;HAFNER;3
\p AUT;Clemens;MÜLLNER;3
\p AUT;Stephan;PFANNERER;3
\p AUT;Valerie;ROITNER;5
\p CRO;Ivana;ANTOLIŠ;5
\p CRO;Ivo;BOŽI\'C;3
\p CRO;Borna;CICVARI\'C;4
\p CRO;Matko;LJULJ;3
\p CRO;Irma;TELAROVI\'C;5
\p CRO;Grgur;VALENTI\'C;3
\p CZE;David;KLAŠKA;2
\p CZE;Jiří;MAREK;4
\p CZE;Van Nhan;NGUYEN;5
\p CZE;Tomáš;PAVLÍK;3
\p CZE;Hana;ŠORMOVÁ;5
```

```
\p CZE;Jan;VAŇHARA;5
\p GER;Bertram;ARNOLD;1
\p GER;Simon;BUCHHOLZ;2
\p GER;Martin;MERKER;2
\p GER;Judit;RECKNAGEL;3
\p GER;Jens;REINHOLD;2
\p GER;Radomyra;SHEVCHENKO;4
\p HUN;András;ÉLES;1
\p HUN;Dávid;FONYŐ;2
\p HUN;Dániel;NAGY;1
\p HUN;Nóra;SZ\H OKE;2
\p HUN;Gergely;SZ\H UCS;1
\p HUN;László;VARGA;2
\p POL;Patryk;DROBI\'NSKI;2
\p POL;Tomasz;KOCIUMAKA;2
\p POL;Karol;KONASZY\'NSKI;2
\p POL;Daniel;MALINOWSKI;2
\p POL;Jakub;O\'CWIEJA;1
\p POL;Damian;ORLEF;3
\p SLO;Eva;BREZNIK;5
\p SLO;Filip;KOZARSKI;5
\p SLO;Matjaž;PAYRITS;4
\p SLO;Tomaž;STEPIŠNIK PERDIH;5
\p SUI;Jürg;BACHMANN;4
\p SUI;Titus;CIESLEWSKI;4
\p SUI;Hrvoje;DUJMOVI\'C;3
\p SUI;Cyril;LAGGER;5
\p SUI;Clemens;POHLE;3
\p SUI;Pascal;SU;3
\p SVK;Ladislav;BAČO;4
\p SVK;Martin;BACHRATÝ;4
\p SVK;Peter;CSIBA;3
\p SVK;Michal;HAGARA;3
\p SVK;Jakub;KONEČNÝ;4
\p SVK;Tomáš;SZANISZLO;4
\end{document}
```

Vlastní práci pak vykoná makro `telo.tex`:

```
\ifcase\typec  \or      %jaká medaile bude vypsána na diplomu
\def\type{GOLD} \or     %tato informace je obsažena v #4
\def\type{SILVER} \or   %význam: 1-Gold, 2-S, 3-B, 4-HM
\def\type{BRONZE} \else \fi
```

```
\font\logoh=pplb8z at 24pt   %připojení písma a
\font\logo=pplb8z at 28pt    %definování přepínačů
\font\pis=pplb8z at 14pt    \font\pisi=pplb8z at 10pt
\font\norm=pplr8z at 14pt   \font\normi=pplr8z at 12pt

%logo olympiády
\leftline{\resizebox{!}{3.8cm}{\includegraphics{loMEMO2}}}
%samotný text na diplomu
\vspace{-2.3cm}
\rightline{\logoh 2\raisebox{7pt}{\pis nd} MEMO}

\vspace{5mm}
\rightline{\pis
\begin{tabular}{l}
4\raisebox{4pt}{\pisi th}\,--\,10\raisebox{4pt}{\pisi th}
September 2008, Olomouc, Czech Republic\\[6pt]
Individual Competition
\end{tabular}}

\vspace{14mm}\vfill
\begin{center}
{\logo \name \kern10pt \surname}

\vspace{5mm}\vfill
{\pis participated in the}

\vspace{-1mm}\vfill
{\logo 2\raisebox{9pt}{\pis nd} MEMO}

\vspace{4mm}
{\pis (Middle European Mathematical Olympiad)}

\vspace{4mm}\vfill
\ifnum\typec<5 {\pis and was awarded a} \fi

\vspace{-2mm}\vfill
\ifnum\typec<4 {\logo \type \kern10pt MEDAL} \fi
\ifnum\typec=4 {\logo Honorary Mention} \fi
\end{center}

\input konec.tex
\endinput
```

Makro `konec.tex` obsahuje pouze podpisy tehdejších pořadatelů, které jsou na všech diplomech stejné, vyřešíme je tedy zvlášť, což umožňuje rychlou úpravu všech jejich výskytů:

```
\vspace{10mm}\vfill\vfill
\leftline{\begin{tabular}{c}
\norm Doc. RNDr. Jaromír Šimša, CSc.\\
\normi Chairman of the Czech MO Committee
\end{tabular}
\hfill
\begin{tabular}{c}
\norm Prof. RNDr. Lubomír Dvořák, CSc.\\
\normi Rector of Palacký University, Olomouc
\end{tabular}}
\endinput
```

4.1.2. Tisk diplomů soutěže družstev

V tomto případě vypadá LaTeXovský dokument takto:

```
\documentclass[a4paper,12pt]{article}
\usepackage[cp1250]{inputenc}
\usepackage[dvips]{graphics}
\usepackage{czech}

\tabcolsep0pt
\def\thepage{}

\voffset-1in \hoffset-12mm
\textwidth16cm \textheight9.8in

\begin{document}

\newcount\typec           %definování parametrů pro družstva
\def\p#1;#2;#3;#4;#5;#6;#7;#8{
\def\state{#1}
\def\1{#2} \def\2{#3} \def\3{#4} \def\4{#5} \def\5{#6}\def\6{#7}
\typec=#8
\input teloD.tex \clearpage
}

\p AUSTRIA;Felix DRÄXLER;Adrian FUCHS;Johannes HAFNER;
```

```
Clemens MÜLLNER;Stephan PFANNERER;Valerie ROITNER;4
\p CROATIA;Ivana ANTOLIŠ;Ivo BOŽI\'C;Borna CICVARI\'C;Matko LJULJ;
Irma TELAROVI\'C;Grgur VALENTI\'C;4
\p CZECH REPUBLIC;David KLAŠKA;Jiří MAREK;Van Nhan NGUYEN;
Tomáš PAVLÍK;Hana ŠORMOVÁ;Jan VAŇHARA;4
\p GERMANY;Bertram ARNOLD;Simon BUCHHOLZ;Martin MERKER;
Judit RECKNAGEL;Jens REINHOLD;Radomyra SHEVCHENKO;1
\p HUNGARY;András ÉLES;Dávid FONYŐ;Dániel NAGY;Nóra SZ\H OKE;
Gergely SZ\H UCS;László VARGA;1
\p POLAND;Patryk DROBI\'NSKI;Tomasz KOCIUMAKA;Karol KONASZY\'NSKI;
Daniel MALINOWSKI;Jakub O\'CWIEJA;Damian ORLEF;1
\p SLOVENIA;Eva BREZNIK;Filip KOZARSKI;Matjaž PAYRITS;;
Tomaž STEPIŠNIK PERDIH;;4
\p SUISS;Jürg BACHMANN;Titus CIESLEWSKI;Hrvoje DUJMOVI\'C;
Cyril LAGGER;Clemens POHLE;Pascal SU;4
\p SLOVAKIA;Ladislav BAČO;Martin BACHRATÝ;Peter CSIBA;
Michal HAGARA;Jakub KONEČNÝ;Tomáš SZANISZLO;4

\end{document}
```

Vlastní práci pak vykoná makro `teloD.tex`:

```
\ifcase\typec  \or
\def\type{GOLD} \or
\def\type{SILVER} \or
\def\type{BRONZE} \else \fi

\font\logoh=pplb8z at 24pt
\font\logo=pplb8z at 28pt
\font\pis=pplb8z at 14pt
\font\pisi=pplb8z at 10pt

\font\norm=pplr8z at 14pt
\font\normi=pplr8z at 12pt

\leftline{\resizebox{!}{3.8cm}{\includegraphics{loMEMO2}}}

\vspace{-2.3cm}
\rightline{\logoh 2\raisebox{7pt}{\pis nd} MEMO}

\vspace{5mm}
\rightline{\pis
\begin{tabular}{l}
```

```
4\raisebox{4pt}{\pisi th}\,--\,10\raisebox{4pt}{\pisi th}
September 2008, Olomouc, Czech Republic\\[6pt]
Team Competition
\end{tabular}}

\vspace{14mm}\vfill
\begin{center}
{\pis The team representing}

\vspace{10mm}

{\logo \state}

\vspace{5mm}

\hbox to \hsize{\hfill\resizebox{8pt}{34pt}{\logo (}\hspace{4pt}
\raise20pt\hbox{\begin{tabular}[t]{c@{\hspace{10pt}}c@{\hspace{10pt}}c}
\norm \1 & \norm \2 & \norm \3 \\[2pt]
\norm \4 & \norm \5 & \norm \6
\end{tabular}}
\hspace{4pt}\resizebox{8pt}{34pt}{\logo )}\hfill}

\vspace{5mm}\vfill
{\pis participated in the}

\vspace{-1mm}\vfill

{\logo 2\raisebox{9pt}{\pis nd} MEMO}

\vspace{4mm}

{\pis (Middle European Mathematical Olympiad)}

\vspace{4mm}\vfill
\ifnum\typec<4 {\pis and was awarded a} \fi

\vspace{-2mm}\vfill

\ifnum\typec<4 {\logo \type \kern10pt MEDAL} \fi
\end{center}

\input konec.tex
\endinput
```

4.1.3. Ukázka výsledného tisku

2nd MEMO

4th – 10th September 2008, Olomouc, Czech Republic
Individual Competition

Adrian FUCHS

participated in the

2nd MEMO

(Middle European Mathematical Olympiad)

and was awarded a

Honorary Mention

Doc. RNDr. Jaromír Šimša, CSc.
Chairman of the Czech MO Committee

Prof. RNDr. Lubomír Dvořák, CSc.
Rector of Palacký University, Olomouc

2nd MEMO

4th – 10th September 2008, Olomouc, Czech Republic
Team Competition

The team representing

HUNGARY

(András ÉLES Dávid FONYÓ Dániel NAGY
Nóra SZŐKE Gergely SZŰCS László VARGA)

participated in the

2nd MEMO

(Middle European Mathematical Olympiad)

and was awarded a

GOLD MEDAL

Doc. RNDr. Jaromír Šimša, CSc.
Chairman of the Czech MO Committee

Prof. RNDr. Lubomír Dvořák, CSc.
Rector of Palacký University, Olomouc

Závěr

Nyní bych ráda zrekapitulovala cíle, které jsme si stanovili v úvodu a shrnula, zda bylo těchto cílů dosaženo.

Prvním a hlavním cílem bylo vytvořit v TEXu databázi bez databázového systému, což se nám povedlo. Vytvořili jsme databáze jak pro jednotlivce, tak pro skupiny lidí. Jsme schopni vytvářet rozmanité databáze a pracovat s nimi. Konečným výstupem byl v našem případě tisk diplomů účastníků matematické soutěže. Na stejném principu bychom mohli např. tisknout hromadně i dopisy, ke kterým bychom si samozřejmě mohli vytisknout i obálky.

Dalším cílem práce bylo srozumitelně popsat způsob vytvoření textové databáze a práce s ní — tj. aby práce mohla dále posloužit jako učební pomůcka. Zda jsme tomuto cíli dostáli či ne, musí posoudit jiní. Troufáme si ale říci, že v této oblasti je každá příručka velkým přínosem, neboť díla podobného obsahu prakticky neexistují.

Posledním (spíše mým osobním) cílem bylo seznámit se a naučit se pracovat se systémem TEX, protože bych i nadále chtěla prezentovat své výsledky v této formě. Myslím si, že tomuto cíli jsem dostála. TEX ovládám na dobré uživatelské úrovni a díky této práci jsem prostudovala řadu publikací, ve kterých lze nalézt odpovědi i na ty nejzapeklitější otázky.

Literatura

[1] Knuth, D. E.: The TEXbook. Addison–Wesley, Reading, 1986.

[2] Kopka, H., Daly, P.: LATEX: podrobný průvodce. Computer press, Brno, 2004.

[3] Olšák, P.: TEXbook naruby. Konvoj, Brno, 2001.

[4] Olšák, P.: Databáze bez databázového programu. Zpravodaj Československého sdružení uživatelů TEXu **3/94** (1994), 118–126.

[5] Oetiker, T.: Ne příliš stručný úvod do systému LATEX 2_ε.

[6] Rybička, J.: LATEX pro začátečníky (3. vydání). Konvoj, Brno, 2003.

[7] www.cstug.cz