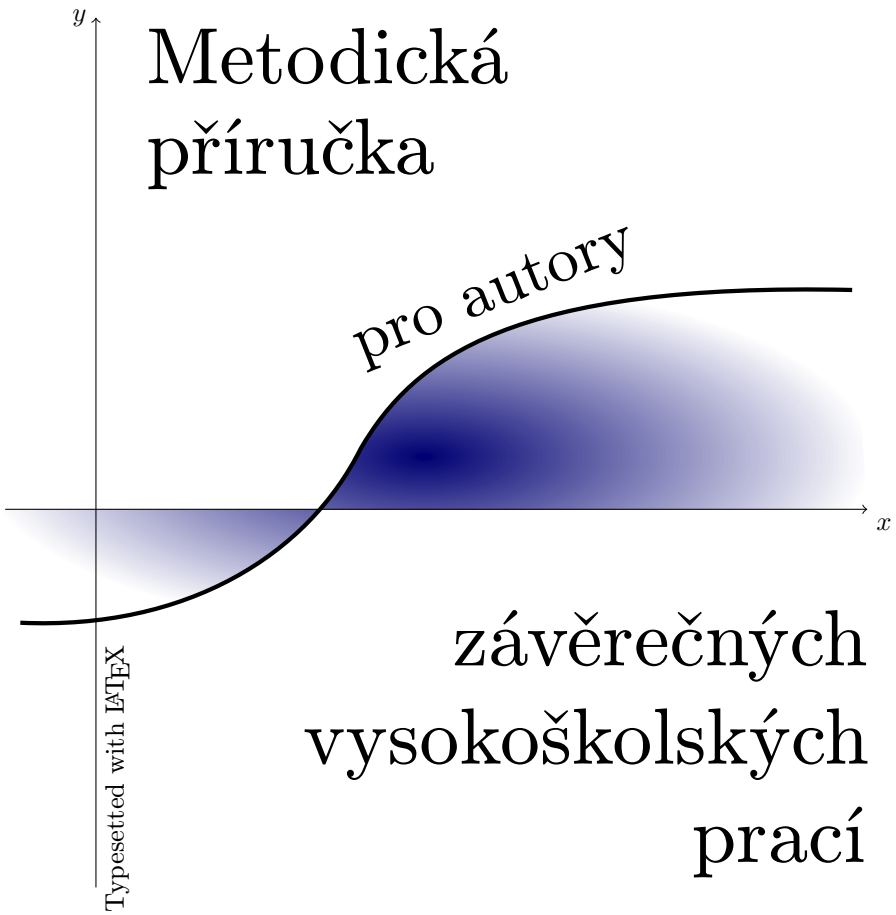
evropský sociální fond v ČR
EVROPSKÁ UNIE
MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY
OP Vzdělávání pro konkurenceschopnost
Slezská univerzita v Opavě
INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Šárka Vavrečková



Metodická příručka závěrečných vysokoškolských prací

Ústav informatiky
Filozoficko-přírodovědecká fakulta v Opavě
Slezská univerzita v Opavě

Opava, poslední aktualizace 8. září 2023

Tato inovace předmětů *Seminář k bakalářské práci, Seminář k diplomové práci* je spolufinancována Evropským sociálním fondem a Státním rozpočtem ČR, projekt číslo CZ.1.07/2.2.00/28.0014, „Interdisciplinární vzdělávání v ICT s jazykovou kompetencí“.

SLEZSKÁ UNIVERZITA V OPAVĚ
Filozoficko-přírodovědecká fakulta v Opavě

BAKALÁŘSKÁ PRÁCE

Opava 2023 Jan Novák

SLEZSKÁ UNIVERZITA V OPAVĚ
Filozoficko-přírodovědecká fakulta v Opavě

Jan Novák

Studijní program: Moderní informatika
Specializace: Informační a komunikační technologie

Metodika k závěrečné práci

Methodics of Thesis

Bakalářská práce

Opava 2023

Vedoucí bakalářské práce:
RNDr. Šárka Vavrečková, Ph.D.

Abstrakt

Text abstraktu v češtině. Rozsah by měl být 50 až 100 slov. Abstrakt není cíl práce, zde stručně popište, co čtenář má na následujících stránkách očekávat. Typické formulace: „V práci se zabýváme...“, „Tato bakalářská práce pojednává o...“, „součástí je“, „je provedena analýza“, „praktickou částí práce je aplikace xxx“ ... Prostě napište stručný souhrn či charakteristiku obsahu práce.

Klíčová slova

Napište 5–8 klíčových slov v českém jazyce (v jednotném čísle, první pád atd.), měla by vystihovat téma práce. Slova oddělujte čárkou. Snažte se vystihnout nejdůležitější pojmy vystihující práci.

Abstract

Anglická verze abstraktu by měla odpovídat české verzi, třebaže nemusí být úplně doslova. Když nutně potřebujete automatický překlad, použijte raději
https://www.deepl.com/cs/translator,
je lepší než Google Translator. Není nutno překládat doslova.

Keywords

Anglická obdoba českého seznamu klíčových slov.

Čestné prohlášení

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškerou literaturu a další zdroje, z nichž jsem při zpracování čerpal, v práci řádně cituji a jsou uvedeny v seznamu použité literatury.

V Opavě dne 8. září 2023

.....

Jan Novák

Poděkování

Rád bych poděkoval za odborné vedení, rady a cenné poznatky k danému tématu vedoucímu práce ..... Také bych rád poděkoval mé rodině a přátelům za podporu a pomoc během mého studia.

Obsah

Úvod

V tomto dokumentu jsou naznačeny možnosti třídy dokumentu *vdocdiplcz*. Tato třída je založena na třídě *article* a načítá balíčky *ifthen* a *fancyhdr*, které jsou běžnou součástí distribucí LaTeXu.

Výchozí nastavení třídy respektují Metodický pokyn pro úpravy, zveřejňování a ukládání závěrečných prací posluchačů Slezské univerzity v Opavě, Filozoficko-přírodovědecké fakulty v Opavě (z roku 2010). V jednotlivých kapitolách dokumentu jsou popsány možnosti změny výchozích nastavení (formát, text prohlášení, poděkování, apod.), běžné požadavky na typografickou kvalitu závěrečné práce a především zde najdeme úvod do matematické sazby a tvorby odborných textů.

Tento dokument je sice míněn také jako ukázka formátování závěrečné VŠ práce, ale ve skutečnosti se jedná spíše o metodickou příručku a některé prvky z tohoto dokumentu nedoporučuji v závěrečné práci používat. Jde především o barevné orientační symboly v poznámkách na okraj, které se zobrazují vlevo od textu. Symbol upozorňuje na pojem či postup, kterému je vhodné věnovat zvýšenou pozornost, symbol značí kód v LaTeXu.

Konfigurační informace dokumentu mohou být uloženy buď do hlavního souboru s příponou .TEX, a nebo do souboru s příponou .CFG pojmenovaného stejně jako hlavní soubor (tento soubor je během překladu načítán a vše, co v něm je, bude zpracováno stejně jako kdyby se to nacházelo před příkazem `\begin{document}` v hlavním souboru).

Předpokládá se, že autor práce nebude vše „mastit“ do jediného souboru, ale jednotlivé kapitoly umístí do samostatných souborů s příponou .TEX. Jednou z výhod tohoto postupu je také to, že když bude třeba přeuspořádat hlavní kapitoly, jednoduše změníme pořadí maker `\include{...}` pro jednotlivé kapitoly v hlavním dokumentu.

Nastavení češtiny: upravte dle svého (zde je použit balíček *czech*, tj. případně nahraďte makro `\usepackage{czech}` tím, co používáte).

Pro překlad doporučuji použít *pdflatex*, popřípadě *pdfcslatex*. Tento dokument byl přeložen takto:

```
pdfcslatex --tcx=cp1250cs -interaction=nonstopmode sablona_ukazka.tex
```

1 Manuál autora závěrečné práce

V této kapitole si projdeme „procesní“ stránku vytváření závěrečné práce od jejího zadání až k obhajobě a státní zkoušce. Uvedené informace jsou platné pro studenty Filozoficko-přírodovědecké fakulty Slezské univerzity v Opavě, pro jiné fakulty či vysoké školy budou samozřejmě platit trochu jiné podmínky.

1.1 Zadání práce

Zadání práce *sestavuje především budoucí vedoucí* této práce, nicméně student může spolupracovat a s vedoucím se domluvit o tématu, směřování práce a konkrétních krocích, které je nutné v rámci plnění cílů zadání provést. Už ze zadání práce by mělo být zřejmé, co konkrétně má student dělat, co má být výstupem, v jaké kvalitě a rozsahu.

Zadání má být formulováno tak, aby nebylo příliš obecné, ale ani příliš konkrétní, aby zůstal určitý manévrovací prostor v případně nepředvídatelných problémů.

Téma má být ve zvládnutelném rozsahu, ale na druhou stranu i na dostatečné odborné výši. Pokud by bylo příliš těžké či hůře časově zvládnutelné, měl by student problém se splněním zadání, naopak pokud by téma bylo příliš „jednoduché“ – triviální, pak by student pro změnu měl vážný problém při obhajobě práce. Například témata typu „vytvořte uživatelskou příručku pro MS Word“ nejsou vhodná z druhého důvodu (MS Word se učí na středních školách, navíc takových příruček existuje na webu spousta).

Jak si najít vedoucího práce s vhodným tématem: vedoucího vybíráme podle typu tématu, na kterém chceme pracovat (například pokud chceme téma týkající se počítačových sítí, půjdeme za některým vyučujícím předmětů jakkoliv souvisejících s počítačovými sítěmi). Nezanedbatelnou roli však hraje i zkušenost a „pověst“ daného budoucího vedoucího.

Na vedoucího jsou kladeny i určité formální požadavky, například co se týče dosaženého stupně vzdělání a titulu. Vyučující obvykle vědí, které typy prací mohou vést. Vedoucím práce může být i externí odborník z praxe, pak není tak důležité, jakého vědeckého či akademického titulu dosáhl (nicméně určitá spodní hranice by

měla existovat). V případě externího vedoucího by měl mít student i interního konzultanta, protože externisté obvykle nemají dostatečný přehled o formálních náležitostech závěrečných prací.

Postup je v souhrnu následující:

1. Student se domluví s budoucím vedoucím práce předně na tom, že tento vedoucí bude vést jeho práci, dále na samotném tématu, znění zadání práce a především jakémsi počátečním harmonogramu (nenechávat vše na poslední chvíli).
2. Domluvené zadání vloží vedoucí ve spolupráci se studentem do informačního systému.

1.2 Jak postupovat při psaní práce

1.2.1 Přípravná fáze a spolupráce s vedoucím práce

Předem si *prohlédněte několik již obhájených prací* (ne moc starých, ideálně čerstvě obhájených). Vedoucí práce by mohl doporučit vhodné práce pro inspiraci (s podobným typem tématu). Všimněte si členění práce, toho, jak co má vypadat (titulní strany, kapitoly různých úrovní s číslováním, obsah, seznam literatury, apod.), jaký je rozsah práce, co má být v analytické či teoretické části, jak je sestavena praktická část, jakým způsobem jsou vkládány obrázky a tabulky, jak je na ně odkazováno, jak se odkazuje do seznamu literatury, jak jsou formátovány citace, atd.

Systematický přístup již od začátku může ušetřit hodně práce a času. Takže si nejdřív rozplánujeme, co vše chceme udělat, jak a kam o tom budeme psát, kolik času zřejmě bude potřeba věnovat jednotlivým částem (tj. sestavíme si *osnovu a harmonogram*).

V harmonogramu bychom měli počítat i s *konzultacemi s vedoucím práce*. Jedná se o to, aby byl student včas a řádně usměrňován – aby nic důležitého nepřeskočil nebo naopak aby se zbytečně dlouze nerozepisoval o něčem, co do práce nepatří nebo co stačí popsat krátce.

Existují studenti, kteří svou práci zašlou vedoucímu k nahlédnutí až pár týdnů před očekávaným odevzdáním práce, což je obrovská chyba: může se stát, že zatímco student svou práci považuje za naprosto perfektní, vedoucí bude mít zcela opačný

názor. Na druhou stranu neočekávejte, že svého vedoucího nadchnete, když mu budete práci posílat co dva týdny během celého půl roku bez vyznačení změn, které jste udělali (nutit číst svého vedoucího co dva týdny několik desítek stránek, to doopravdy nejde).

1.2.2 Struktura práce

Tento dokument sám o sobě naznačuje, jakou strukturu by závěrečná práce měla mít. Ujasníme si, co patří do jednotlivých částí.

V *Úvodu* píšeme o tom, co je cílem práce, můžeme vyložit základní myšlenku, důvod, proč je zpracování tohoto tématu důležité a užitečné, co bude v rámci práce vytvořeno. Součástí úvodu může být méně formální vysvětlení některých pojmů, které se vyskytují v zadání a na kterých stojí celé téma, aby čtenář hned v Úvodu pochopil, co má dále očekávat a jaké znalosti jsou u něj předpokládány. Může být zde také krátké shrnutí obsahu práce (co se čtenář v jednotlivých kapitolách či skupinách kapitol dozví), a jestliže jsme vytvořili něco vlastního, upozorníme čtenáře na ten fakt a sdělíme mu, kde to najde (třeba na přiloženém CD). Pokud jsme v práci využívali některé ne zcela běžné prostředky nebo plnění cílů práce probíhalo v konkrétním specifickém prostředí, taktéž bychom na to měli v Úvodu upozornit.

Následuje jedna či více kapitol *analytické (popř. teoretické) části*. Pokud je to nutné, zde se věnujeme vysvětlení potřebných pojmů a postupů, které dále využijeme. Jestliže máme volit mezi různými prostředky, nástroji, programy, postupy apod., zde je popíšeme, srovnáme a vysvětlíme, proč jsme tak volili. Obecně platí, že čím vyšší typ práce (v rozsahu od bakalářské po disertační), tím rozsáhlejší by tato část měla být. Co sem skutečně napíšeme, záleží samozřejmě na zaměření práce (něco jiného můžeme očekávat u práce s matematicky orientovaným tématem než třeba u práce zaměřené na programování robotů). Pokud popisujeme různé nástroje (včetně aplikací) určené pro tentýž účel, vždy bychom měli přidat nějaký souhrn či zhodnocení vhodnosti pro daný účel. Pokud je srovnání hlavním cílem práce, pak bychom měli uvést i *metodiku* hodnocení a srovnání (tj. popis kritérií, podle kterých hodnotíme, jejich váhu, způsob, jakým jsme došli k publikovanému výsledku).

Praktická část je pouze u takových prací, kde se s nějakou praktickou částí počítá (například popis sestrojení zařízení, popis konfigurace či zapojení sítě, popis

naprogramované aplikace s popisem postupu jejího naprogramování, ...). Rozhodně nevolíme názvy kapitol „Teoretická část“ a „Praktická část“, tím se automaticky okrádáme o jednu úroveň kapitol. Přímo z názvů kapitol by mělo být zřejmé, do které části patří.

V *Závěru* shrnujeme dosažené cíle a výsledky, uvádíme, co vše bylo v rámci plnění cílů práce vytvořeno, především zdůrazňujeme, co vše jsme osobně udělali, co je naše práce (když je v elektronické příloze něco, co jsme vytvořili, taky se tím pochlubíme). Závěr slouží k tomu, aby si čtenář (zejména oponent) ujasnil, jak moc náročná, kvalitní a rozsáhlá činnost s prací souvisela.

Seznam použité literatury tvoříme „za běhu“. Kdykoliv použijeme některý zdroj informací, hned ho v seznamu uvedeme. Je to důležité i z toho důvodu, že citované texty a různé objekty musí být řádně označeny svým zdrojem, který se nám dodatečně nemusí hledat zrovna snadno.

1.2.3 Hlavní zásady

Shrneme si zásady, které bychom měli při psaní práce dodržet:

- Nepodceňujte *přípravu* – prohlédnutí několika obhájených prací, osnova, harmonogram.
- *Konzultace* s vedoucím práce. Pokud opakovaně zasíláte text práce ke zkontrolování, vyznačte oblasti, které jste změnili, aby vedoucí nemusel číst celou práci znovu. Snažte se opravami nespáchat další chyby.
- *Nic neodkládejte!* Nepočítejte s tím, že dva měsíce před odevzdáním budete mít dostatek času. Nebudete. Nejméně měsíc předem byste měli mít práci ve stavu, který je už víceméně finální, počítejte s tím, že vedoucí práce i v této „finální“ práci najde hodně míst, kde je třeba provést změny. Nezapomeňte, že vytisknutí a svázání práce může trvat několik dnů.
- Naučte se *pravopis*. Závěrečná práce je vlastně odbornou prací, a dělat hrubky v odborných pracích je ostuda. Co se týče abstraktu v angličtině, nespoléhejte na Google Translate či podobné nástroje, vygenerovaný text je kostrbatý a často i nesprávný (hlavně co se týče použití odborných výrazů). Pravopis máte pilovat na střední škole, před maturitou.

- V textu používáme *autorský plurál* (v tomto dokumentu to neplatí, nejde o závěrečnou práci). Znamená to, že používáme formulace typu „dále popíšeme“, „zde zadáme“, „musíme uložit“, „vidíme“, „další informace najdeme v“, apod. Případně lze autorský plurál obejít formulacemi typu „je možné provést“, „program dovoluje“, „je třeba zadat“, atd. Zásadně neoslovujeme čtenáře (opět to neplatí pro tento dokument). Autorský plurál nemusíme použít například v Úvodu tam, kde píšeme o své vlastní motivaci k psaní práce (co mě vedlo k volbě tohoto tématu), případně v Závěru při zhodnocení přínosu práce.
- *Seznam použité literatury* sestavujeme průběžně.
- Veškeré přejaté texty, obrázky, statistické údaje, obsah tabulek, apod. musíme *citovat* (tj. přidáme odkaz na zdroj, je popsáno dále).

Nedodržení těchto zásad se může hodně vymstít – buď tím, že se nám najednou nahromadí tolik práce, že nebudeme stíhat, nebo velmi kritickým posudkem jak oponenta, tak dokonce vedoucího práce (což je opravdu velká ostuda, a občas se to bohužel stává).

1.3 Jak odevzdat závěrečnou práci

Znovu připomínám, že práce by měla být v téměř finální podobě alespoň měsíc před termínem odevzdání, počítáme také několik dní na vytisknutí a svázání. Ovšem je možné se domluvit v příslušné firmě na tom, že desky vyrobí předem, k čemuž potřebují znát přibližný počet listů kvůli šířce hřbetu a samozřejmě text, který se má na desky vysázet. Pak svázání obvykle trvá jeden den.

Veškeré termíny na FPF SU v Opavě související se závěrečnými zkouškami včetně termínu odevzdání závěrečné práce jsou v harmonogramu státních závěrečných zkoušek, který kromě jiného najdete na https://www.slu.cz/fpf/cz/sostatnizaverecnezkousky

Závěrečná práce se odevzdává na sekretariátu příslušného ústavu či katedry. Zároveň s prací student odevzdává i *přihlášku ke státní závěrečné zkoušce*, ale ta pro změnu patří na studijní oddělení.

Se sekretářkou ústavu se domluvte **předem** na zpřístupnění archivu práce v informačním systému (musí ho zpřístupnit vám, abyste tam mohli nahrát soubor

se svou prací, a taky vedoucímu, aby tam mohl nahrát svůj posudek, oponentovi odklepne přístup vedoucí).

Práce se odevzdává elektronicky do archivu závěrečné práce, a pak v jednom výtisku v pevných deskách na sekretariát ústavu/katedry. Sekretářka zkontroluje, zda tištěná verze odpovídá verzi elektronické, v IS odklepne odevzdání a až potom se k obsahu archivu práce dostane vedoucí a oponent práce. Proto byste neměli odevzdávat tištěnou verzi se zpožděním, aby dotyčné osoby mohly začít s psaním posudku. Čím dřív budete mít posudek k dispozici, tím lépe.

1.4 Obhajoba práce a státnice

1.4.1 Termíny a přihlášení

V harmonogramu na výše uvedené adrese najdeme obecný termín, ve kterém se konají státní závěrečné zkoušky, konkrétní den je upřesněn obvykle několik týdnů předem (podle toho, kolik studentů se přihlásí, aby bylo jasné, kolik státnicových dnů bude třeba vypsat).

Tento den je obvykle uveden i na stránkách příslušného ústavu či katedry, například pro Ústav informatiky dotyčnou informaci najdeme v části *Studium* ⇒ *Státní závěrečné zkoušky*. Po upřesnění se tyto termíny objeví v IS jako zkouškové termíny.

Pokud se z vážného důvodu nemůžeme ke státní zkoušce dostavit či máme zpoždění, je třeba informovat sekretářku a domluvit se na řešení. Jestliže se včas a řádně neodhlásíme, termín nám propadá. Je jediná možnost, jak tomu zabránit, když jsme odhlášení z vážných důvodů nestihli — poslat dodatečně omluvu s důkladným zdůvodněnín na studijní oddělení, pokud možno například s dokladem od lékaře, který dokazuje, že student nebyl schopen přijít, třeba proto, že byl v komatu.

V harmonogramu si všimneme, v kolik hodin začínáme my, ale ten čas berme jen orientačně. Rozhodně musíme být na místě ještě před začátkem na slavnostní zahájení obhajoby a státní závěrečné zkoušky. Navíc ještě před zahájením bychom měli zajistit nahrání své prezentace na připojený počítač.

1.4.2 Jak se připravit

V státnicovém dni se zpravidla nejdřív konají obhajoby závěrečných prací a následně proběhnou samotné státní zkoušky, obvykle vše v jednom dni. Obhajoba práce sestává z těchto kroků:

1. Student představí svou práci s využitím prezentace.
2. Jsou čteny posudky vedoucího a oponenta práce.
3. Student má možnost reagovat na posudky (například zodpoví otázky kladené v posudcích, vysvětlí vytýkané nejasnosti a chyby).
4. Proběhne diskuse, ve které student zodpoví otázky členů komise.

Zkoušení u státní závěrečné zkoušky má už jednodušší strukturu – student si z obálky vytáhne otázky (za každý státnicový předmět/okruh jednu otázku), dostane čas na přípravu včetně možnosti dělat si poznámky a následně „dostane slovo“. V ideálním případě povídá sám, v méně ideálním případě povídá podle otázek zkoušejícího.

Předem se připravíme následovně:

- Vytvoříme *prezentaci* své závěrečné práce, podrobnosti níže.
- Své vystoupení včetně ovládání prezentace si *nacvičíme*. Měli bychom mít jasno v tom, co kdy budeme říkat, a také kolik toho asi tak stihneme říct ve vymezeném čase (pozor, čas podle harmonogramu zahrnuje i čtení posudků a diskusi). Je třeba si uvědomit, že ve stresu člověk obvykle mluví pomaleji než jindy.
- Opatříme si *posudky* vedoucího a oponenta. Přečteme si je a připravíme si odpovědi na otázky a výtky.
- *Naučíme se na státnice*. To není triviální záležitost! Uvědomte si, že hlavně tehdy, pokud jste se učili stylem „aby ty body nějak vyšly“, vědomosti se z hlavy vykouřily a je nutné je tam znovu dostat. Většina studentů potřebuje minimálně měsíc před státnicemi intenzivní učení. Upozorňuji, že Murphyho zákony se bohužel projevují právě při takovýchto příležitostech.
- Ke státnicím je třeba přijít ve *vhodném oblečení*.

U státnic se prosím neznervózňujte navzájem, je to bezohledné. Existují studenti, kteří během čekání ostatním vykládají, jak jsou ty otázky vlastně jednoduché, že

se učili tak kolem jednoho týdne, atd. Volte spíše uvolněnější a pozitivní témata konverzace (počasí, koníčky, pomlouvání vyučujících, apod.).

1.4.3 Jak má vypadat prezentace závěrečné práce

Prezentaci můžeme vytvořit v PowerPointu, OpenOffice.org či Libre Office Impress, případně můžeme využít některou třídu LaTeXu (většinou beamer nebo prosper). Pro jistotu si kromě souboru ve formátu příslušného nástroje připravíme i variantu v PDF.

K obhajobě přijdeme předem (před zahájením), zkopírujeme prezentaci na předváděcí počítač a *vyzkoušíme* (může nastat problém s verzemi, hlavně u PowerPointu) – když je prezentace nefunkční nebo „rozhozená“, dotyčného to vždy rozhodí.

Nebudeme zde probírat konkrétní postupy, jen probereme hlavní zásady:

- S grafickou úpravou snímků a přechody to nepřehánějte, nejde přece o promítání fotek z dovolené. Snímky by měly být vkusně upraveny, klidně i v barvách, jen všeho moc škodí. Přechody mezi snímky by neměly být „časově náročné“.
- Na prvním snímku by se posluchači měli dozvědět, jak se práce nazývá, kdo je autorem a kdo vedoucím práce.
- Na jeden snímek neumísťujeme příliš mnoho textu. Nenuťte své posluchače být zároveň pilnými čtenáři. Text sice ano, ale jen velmi stručně ve formě bodů, které ústně rozvádíme. Text berte jako záchytné body jak pro posluchače, tak i pro sebe. Pokud obhajobu pojmete jako čtení obsahu snímků, přijdete o cenné body v hodnocení a pokud navíc jsou posudky velmi kritické, zvýší se pravděpodobnost, že si na příštím termínu obhajobu zopakujete.
- Do prezentace patří obrázky, tabulky (opět neplýtváme textem, ani v buňkách tabulky), nákresy, diagramy a jiný ilustrační materiál. Sekundárním úkolem je oživení prezentace (posluchači by se neměli nudit, ale taky to nepřehánějte).

- Primárním cílem je představit svou práci a především sdělit, co vše autor sám vytvořil, *pochválit se*. Berte v úvahu, že většina členů komise o této práci slyší poprvé, tedy potřebuje dostat nějaké (pozitivní) informace ještě před

(někdy negativními) informacemi z posudků. Nevysvětlujte základní pojmy, které členové komise zřejmě už budou znát, soustřeďte se na samotnou práci.

1.4.4 Preentace v beameru

Pokud se rozhodnete použít LaTeX a třídu *beamer*, první řádek dokumentu může vypadat takto:

```
\documentclass[t,compress,aspectratio=1610]{beamer}
```

Vlastnost *aspectratio* udává poměr stran snímků. Zde použité číslo 1610 znamená poměr stran 16:10. Kdybychom chtěli širokoúhlý 16:9, pak použijeme číslo 169. Výchozí hodnota (pokud tuto vlastnost vůbec nepoužijeme) je 4:3, což dnes už moc neodpovídá moderním projektorům.

Dále můžeme nastavit základní vzhled prezentace, například:

```
\usetheme{Singapore}
\usefonttheme[onlymath]{serif}
\setbeamercovered{transparent=15}
```

To znamená, že se použije šablona Singapore (šablon je spousta, stačí se poptat Googlu na třídu beamer), pro běžný text na snímcích bude použito bezpatkové písmo, ale pro matematické prostředí se použije patkové, poslední řádek nastavuje úroveň transparentnosti rámců.

Podobně jako v běžném dokumentu nastavíme titulek dokumentu, autora apod. Snímek můžeme buď vytvořit „ručně“, nebo využijeme následující:

```
\title[Autor: Název práce]{Název závěrečné práce}
\subtitle{Bakalářská/diplomová práce}

\author[Jméno autora]{{%
   Autor: Jméno autora\\[.2ex]
   Vedoucí: Jméno vedoucího\\[.2ex]
   Oponent: Jméno oponenta
}}
\institute[]{Ústav informatiky\\
   Filozoficko-Přírodovědecká fakulta\\
   Slezské univerzity, Opava}

\begin{document}
\frame{\maketitle}
```

Makro `\author` může mít kromě povinného parametru (ve složených závorkách) i nepovinný parametr v hranatých závorkách. Ten nepovinný se buď vůbec

nezobrazuje, nebo si můžeme se stylem pohrát a zajistit třeba zobrazování v zápatí (příslušná makra najdete v dokumentaci k balíčku beamer nebo v různých návodech na Internetu).

Jednotlivé snímky jsou uzavřeny v prostředí *frame*, snímek můžeme (nemusíme) členit na bloky. Pokud se rozhodneme používat sekce, bude v záhlaví snímků jejich seznam a posluchači podle nich poznají, jak jste daleko:

```
\section{Název části prezentace}

\begin{frame}{Název snímku}
\begin{block}{Název bloku}
   \begin{itemize}
      \item členové komise chtějí vědět, o co v práci jde, co bylo vaším úkolem
      \item inspirujte se
      \begin{itemize}
         \item zadáním práce sepsaným, když jste s prací začínali
         \item abstraktem, úvodem práce
      \end{itemize}
      \item<2> tato stránka by měla být
      \begin{itemize}
         \item krátká
         \item stručná
         \item výstižná
      \end{itemize}
   \end{itemize}
\end{block}
\end{frame}
```

2 Nastavení údajů o závěrečné práci

2.1 Jak se vyznat v souborech

Celý projekt pro závěrečnou práci se skládá z několika souborů (zde jsou pojmenovány podle ukázkového dokumentu, místo „sablona_ukazka“ si samozřejmě můžete dosadit vlastní název):

`vdocdiplcz.cls` třída dokumentu pro závěrečné práce; nepřejmenovávejte, doporučuji neměnit ani obsah tohoto souboru

`sablona_ukazka.tex` hlavní soubor projektu, obsahuje některá nastavení typická pro konkrétní práci, a také je to rozcestník pro jednotlivé kapitoly

`config.tex` konfigurační soubor obsahující další nastavení typická pro konkrétní práci

`kapitola00_uvod.tex` text úvodní kapitoly, podobně i pro závěr

`kapitola01_nazev.tex` další kapitola; pojmenujte si dle svého uvážení, nezapomeňte pak ale název souboru změnit v hlavním souboru, totéž platí i pro soubory s dalšími kapitolami

`literatura.tex` soubor se seznamem literatury; během psaní své závěrečné práce byste sem měli průběžně dopisovat veškeré zdroje, které jste použili

2.2 Doplnění konkrétních údajů o práci

V hlavním souboru (před příkazem `\begin{document}`) nebo v konfiguračním souboru mají být použita makra pro personalizaci práce, například typ práce, jméno autora, vedoucí práce apod. Máte tyto možnosti:

`\autor{...}` do složených závorek napište své jméno a příjmení, případně i titul, kterého jste již v minulosti dosáhli

`\vedouci{...}` zadejte jméno a příjmení vedoucího práce včetně titulů

`\nazev{...}` zadejte název práce v jazyce českém, musí odpovídat zadání

`\nazevanglicky{...}` zadejte název práce v jazyce anglickém

`\rok{...}` rok *obhajoby* práce

`\typprace{...}` doplňte některý z řetězců „Bakalářská práce“, „Diplomová práce“, „Rigorózní práce“, „Disertační práce“

Dále si ujasněte, co vlastně studujete – je to k nevíře, ale někteří asi nevědí jistě... V informačním systému si najděte informace o sobě a podívejte se, jak se nazývá váš studijní program, a dále jestli je se specializacemi, případně u starších studijních programů je i obor. Vyplňte svůj studijní program, a pak (pokud tedy existuje) specializaci nebo obor.

`\studijniprogram{...}` zadejte název vašeho studijního programu

`\studijnispec{...}` pokud studujete v programu se specializacemi, zadejte specializaci

`\obor{...}` napište název studijního oboru (na Ústavu informatiky už jen Aplikovaná informatika ve stejném studijním programu)

Další možnosti již vyžadují napsání odstavců textu.

Předně bychom měli vložit český a anglický abstrakt a ke každému klíčová slova. Abstrakty a klíčová slova zadáme pomocí těchto maker:

`\abstrakt{...}` do složených závorek zadáme text českého abstraktu

`\abstraktanglicky{...}` text anglického abstraktu

`\klicovaslova{...}` česká klíčová slova (nejméně 3), oddělíme je čárkou nebo středníkem

`\klicovaslovaanglicky{...}` anglická klíčová slova (přeložíme ta česká)

Jestliže jsou abstrakty příliš dlouhé a na jednu stranu se nevejdou, můžeme vynutit umístění českých a anglických údajů na zvláštní strany, k tomu slouží makro `\dlouhyabstrakt` (alternativně `\longabstract`).

Předefinováním makra `\kopiepodkladu` pak vložíme obrázek s naskenovaným podepsaným zadáním. Například pokud máme zadání uložené v souboru `zadani.png` umístěném ve stejném adresáři jako hlavní soubor, pak tu bude:

```
\renewcommand{\kopiepodkladu}{%
  \noindent
  \begin{picture}(0,0)
  \put(0,-350){\includegraphics[width=\textwidth]{zadani.png}}
  \end{picture}}
```

Míry v makru `\put` a případně rozměry prostředí samozřejmě upravte tak, aby se zadání zobrazilo nějak „vhodně“. Pokud je zadání na více než jedné stránce, vložte postupně víc souborů (na víc stran).

2.3 Úprava textů prohlášení a poděkování

Součástí práce musí být Čestné prohlášení studenta o tom, že práci vypracoval samostatně a uvedl všechny prameny, které při psaní práce použil. Konkrétní znění prohlášení není zcela přesně stanoveno, ale smysl musí být dodržen. Obvyklé znění je toto:

> Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškerou literaturu a další zdroje, z nichž jsem při zpracování čerpal, v práci řádně cituji a jsou uvedeny v seznamu použité literatury.

nebo:

> Prohlašuji, že jsem tuto bakalářskou práci napsal samostatně, pouze za odborného vedení vedoucího práce .... Dále prohlašuji, že veškeré podklady, ze kterých jsem čerpal, jsou uvedeny v seznamu literatury. Nemám závažný důvod proti užití tohoto školního díla ve smyslu §60 Zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

Dámy/slečny samozřejmě budou muset provést úpravu sloves v třetí osobě (vypracovala, čerpala), také je možné přidat třeba větu „Souhlasím s prezenčním zpřístupněním své práce v Ústřední knihovně SU“ (ovšem práce bude zpřístupněna pro prezenční prohlížení i v případě, že tam ta věta nebude).

Výsledné prohlášení zadáme jednoduše pomocí makra

```
\cestneprohlaseni{Prohlašuji, .....}
```

Je vhodné poděkovat svému vedoucímu, případně i jiným lidem (rodině, přátelům, spolupracovníkům, apod.) za pomoc, trpělivost, podporu, povzbuzení, atd. Je to sice nepovinný prvek, ale bylo by nezdvořilé ho nepoužít. K zadání poděkování slouží makro

```
\podekovani{.....}
```

Pokud máte pocit, že není komu děkovat (opravdu?), použijte makro `\bezpodekovani`, aby nebyla vygenerována stránka s poděkováním.

2.4 Struktura hlavního souboru

Hlavní soubor projektu má vcelku jednoduchou základní strukturu:

- deklarace třídy určující hlavní styl celého dokumentu

  `\documentclass{vdocdiplcz}`

- preambule – oblast pro načtení stylových souborů a zadání nastavení platných pro celý dokument (tady můžete vpisovat či měnit nastavení uvedená v sekcích výše)
- hranice mezi preambulí a obsahem dokumentu

  `\begin{document}`

- vytvoření prvních stran dokumentu:

  `\maketitlepages` ..... umístění titulních stran až po poděkování včetně

  `tableofcontents\clearpage` ..... vygenerování obsahu

  `\nastavodstavec` .. nastavení parametrů odstavce platných pro celý dokument

  `\radkovani` nastavení řádkování pro celý dokument (v nepovinném parametru lze určit konkrétní řádkování, výchozí je 1,5)
- hranicí mezi titulními stranami a kapitolami, začátek hlavní části dokumentu

  `\mainmatter`

- kapitoly:

  `\include{název souboru s kapitolou}`

- to, co patří na konec dokumentu:

  `\include{literatura}` ..... seznam použité literatury

  `\listoffigures` ..... nepovinné, seznam obrázků

  `\listoftables` ..... seznam tabulek
- konec dokumentu

  `\end{document}`

To, že umístíme jednotlivé kapitoly do zvláštních souborů, nám dává jednu zajímavou výhodu: během psaní práce si můžeme nechávat překládat třeba jen tu kapitolu, na které zrovna pracujeme (překlad je pak svižnější, nezdržuje nás). Stačí v preambuli (před `\begin{document}`) použít příkaz

`\includeonly{název souboru}`

(případně můžeme zadat několik souborů oddělených čárkou).

2.5 Text práce

Text práce musí být strukturován do (číslovaných) kapitol. Při členění do kapitol zachováváme tato pravidla:

- každá kapitola by měla tvořit logický celek, totéž platí o podkapitolách a kapitolách třetí úrovně,
- pokud je (pod)kapitola příliš dlouhá (řekněme více než dvě stránky, ale hodně záleží na konkrétním obsahu), dále ji členíme, tedy použijeme další úroveň kapitol,
- používáme maximálně tři úrovně kapitol, více úrovní znamená, že jsme nezvládli členění (tj. chce to přestrukturovat),
- souvislý text v rámci (pod/pod)kapitol by neměl být ani příliš krátký, takto oddělovat krátký odstaveček či jednu větu nemá moc smysl.

Úvod a Závěr mají být nečíslované kapitoly, ostatní budou číslované. Forma číslování není v LaTeXu problém (čísla jednotlivých úrovní jsou oddělena tečkou, za číslem poslední úrovně použité v nadpisu se tečka nedělá – i kdyby šlo o úroveň první).

Kapitoly (jejich názvy) zadáváme pomocí maker:

`\section{...}` kapitola první úrovně; pokud použijeme zásadu „co soubor, to kapitola“, pak by v každém souboru s kapitolami měl být právě jeden tento příkaz

`\section*{...}` varianta s hvězdičkou je pro nečíslované kapitoly (Úvod a Závěr), ovšem v Obsahu se taky objeví

`\subsection{...}` kapitola druhé úrovně

`\subsubsection{...}` kapitola třetí úrovně

Pro kapitoly druhé a třetí úrovně sice také existují makra s hvězdičkou pro vložení nečíslované kapitoly, ale v závěrečné práci je nedoporučuji používat (a taky se neobjeví v Obsahu).

2.6 Další možnosti úprav

V následujících kapitolách jsou popsány a na příkladech ukázány možnosti pro sazbu nejběžnějších prvků, se kterými se lze v závěrečných pracích setkat. Pokud to nebude stačit, není problém přidat další balíčky a případně si vytvořit vlastní příkazy a prostředí.

V závěrečné práci se mohou hodit různé speciální symboly (například matematické symboly z ams balíčků, případně symboly z *textcomp*, *pifont*, *dingbat* a dalších). Určitá střízlivost by měla být zachována, přesto se nám může hodit:

- alternativní způsob zápisu tabulek, třeba pro hodně dlouhé tabulky,
- speciální prostředí pro zápis algoritmů,
- speciální prostředí pro zápis teorémů/vět/definic apod.,
- nástroje pro vytváření diagramů, grafů, vektorové grafiky,
- prostředky zvyšující interaktivitu elektronického dokumentu (do IS vkládáme elektronickou verzi práce, případně můžeme vytvořit interaktivní tutoriál či příručku).

Pokud chcete seznam příloh nějak podrobněji propracovat, můžete upravit titulní stránku příloh (je vygenerována přímo před přílohami). Je definována v souboru `config.txt`, konkrétně na jeho konci, tedy stačí pozměnit definici makra `\appendixpage`.

3 Typografie a zápis vkládaných prvků

V této kapitole se podíváme na nejdůležitější typografická pravidla a způsob, jak je dodržet v LaTeXu. Zaměříme se hlavně na to, s čím mívají studenti v závěrečných pracích nejvíc problémů.

3.1 Písmo

V Metodickém pokynu je předepsáno použití písma Computer Modern, které je v LaTeXu výchozí (takže nemusíme nikde nic nastavovat). Pokud bychom chtěli využít písmo *Times*, je to samozřejmě možné (v Metodickém pokynu je předepsáno pro případ, že píšeme v „myšovém“ systému jako je třeba MS Word): stačí do preambule (někam mezi `\documentclass{...}` a `\begin{document}`) přidat makro

`\usepackage{times}`.

Implementace písma *Times* je v LaTeXu mnohem kvalitnější než ve fontu Times New Roman pro Windows, takže text bude v každém případě vypadat úpravněji, než kdybychom použili MS Word nebo OpenOffice Writer.

3.1.1 Vyznačování

Jednou z nejběžnějších chyb při psaní závěrečných prací je špatné vyznačování v textu. Je třeba si uvědomit, že sazba závěrečné práce má být spíše hladká, není to dětská knížka, nic by vysloveně nemělo „bít do očí“. Vyznačit ano, ale nikoliv přehnaně. Probereme si jednotlivé možnosti vyznačení:

- *Kurzíva* – nejvhodnější způsob vyznačování pojmů v textu (z textu dostatečně vystupuje, snadno pak pojem najdeme, ale přitom příliš nebije do očí). Také se používá pro vyznačení citovaného (přejatého) textu. Pokud chceme vyznačit něco, co je uvnitř textu psaného kurzívou, napíšeme to běžným písmem (pravidlo zachování kontrastu). V LaTeXu je nejlepší použít makro `\emph{...}`, které dotyčné pravidlo kontrastu zachovává.

- *Podtržení* – používáme pouze pro odkazy (typicky hypertextové, především odkazy na WWW stránky). Podtržením se vyznačuje na mechanických psacích strojích, protože tam jednoduše není jiná možnost.

- *Tučné písmo* – používá se pro nadpisy a dále například ve slovnících a glosářích pro zvýraznění vysvětlovaného pojmu. Co se nadpisů týče, pokud použijeme pojmenovaný odstavec (viz níže), jedná se taky o nadpis (toho odstavce), tudíž také můžeme použít tučné písmo. Za určitých okolností můžeme použít tehdy, když vysvětlujeme více pojmů s tím, že je uvedeme v seznamu (typicky *description*) vždy na začátku položky (místo „odrážky“) a za tímto pojmem následuje vysvětlení.

  Tučné písmo *není vhodné pro vyznačení pojmu uvnitř textu*, v takovém případě bychom měli použít kurzívu.
- *Neproporcionální písmo* (Typewriter, obdoba Courieru z Windows) – používáme pro vkládaný kód, případně názvy souborů. Taky je možné využít pro hypertextové odkazy.
- *Bezpatkové písmo* – za určitých okolností se může kombinovat s patkovým (ale musíme k tomu mít vážný důvod), například se může využít pro hypertextové odkazy (jak je tomu v tomto dokumentu), pokud nechceme, aby zabíraly příliš mnoho místa.
- *Jiný font* – různé fonty není vhodné kombinovat (častá chyba uživatelů MS Wordu), je to estetické faux pas.
- *Zvýšení rozteče mezi písmeny, prokládání* – používáme jen v naprosto nevyhnutelných případech, nikoliv v závěrečné práci.

Minusky, verzálky, kapitálky – minusky jsou malá písmena, verzálky jsou VELKÁ PÍSMENA, kapitálky jsou „zmenšená“ velká písmena – Písmena, mohou být použity například pro nadpisy nejvyšší úrovně (ale ne v závěrečné práci) nebo příjmení autora citovaného díla.

Pojmenovaný odstavec. Tučné písmo se tedy typicky používá pro názvy kapitol. Svůj název však může mít i odstavec (jako například tento), pak je tučné písmo také na místě. V LaTeXu vytvoříme pojmenovaný odstavec jednoduše pomocí makra

```
\paragraph{Název odstavce.} Text odstavce
```

Tučným písmem se také mohou vyznačovat pojmy ve slovníkovém seznamu (třeba v glosáři, výkladovém slovníku). Ukázku vidíme na straně 21 v sekci o mezerách, seznam je vytvořen pomocí prostředí *description*.

3.1.2 Spojovník, pomlčka, uvozovky a spol.

Studenti si často pletou spojovník s pomlčkou, a dále různé typy mezer. Podíváme se, v jakých situacích který prvek používáme, a také jak ho vytvořit.

Spojovník najdeme přímo na klávesnici, neobklopuje se mezerami. Slouží nejen k rozdělení slova na konci řádku (tj. spojuje dvě části, které byly rozděleny koncem řádku), ale také k připojení podmínkové části slova „-li“ (bude-li, vidí-li) a ke spojení více slov názvu jediného kompaktního objektu (česko-anglický slovník, Ostrava-jih).

Pomlčka je delší než spojovník, na klávesnici ji nenajdeme. V LaTeXu pomlčku vyjádříme jednoduše jako dva spojovníky hned za sebou (`--`), přeloží se jako pomlčka. Pomlčka se používá ve dvou různých případech:

- jako náhrada čárky v souvětí, v tom případě se obklopuje mezerami (uvedl – pro upřesnění – celkové náklady),
- pro vyjádření rozsahu či určitého vztahu, a pak se mezerami neobklopuje (trvá 10–15 minut, zápas Sparta–Slavia, trasa Praha–Prčice). Tento případ poznáme také podle toho, že pomlčku zde můžeme nahradit některým slovem (například až, proti, z … do, mezi … a).

Uvozovky v českém textu odlišujeme vstupní (levé) a výstupní (pravé). Vstupní se píšou dole a mají tvar devítek, výstupní se píšou nahoře a mají tvar šestek. MS Word obvykle uvozovky takto formátuje automaticky, v LaTeXu to některé editory umí samy, obvykle také můžeme použít makro `\uv{text}`. Pokud toto makro neexistuje, můžeme si je vyrobit sami:

```
\newcommand{\uv}[1]{\quotedblbase #1\textquotedblleft}
```

Makra `\quotedblbase` a `\textquotedblleft}` se hodí i v případě, že chceme (z důvodu citování) dát do uvozovek blok textu, na jehož konci je seznam a pravá uvozovka tedy musí být na konci seznamu. LaTeX by se v takovém případě při použití `\uv{...}` vzbouřil, protože levá uvozovka by byla vně prostředí *itemize* (před ním), kdežto pravá uvozovka uvnitř (na konci poslední `\item`).

Kromě těchto „oblých uvozovek“ lze použít ještě »úhlové uvozovky«. V české typografii úhlové uvozovky míří k uvozovanému textu, zatímco ve francouzštině, kde se hodně používají, míří obráceně – směrem od uvozovaného textu.

Před tečkou neděláme mezeru, za ní ano, mezi tečku a pravou závorku však mezeru nedáváme (podobně pro čárku a středník), a totéž platí i pro uvozovky.

3.1.3 Mezery

Není mezera jako mezera. Jaké typy mezer máme a jak je vložíme:

Nezlomitelná mezera je mezera, která nesmí být na konci řádku. Používá se především za jednoznakovými předložkami a spojkami (k, z, v, o, a, i, ...), výjimkou je spojka „a“ v úzkém sloupci. Dále tuto mezeru vkládáme mezi iniciálu jména a příjmení, případně několik iniciál, a mezi slovo „strana“ (příp. varianty včetně zkratky „str.“) a číslo strany.

V LaTeXu vložíme nezlomitelnou mezeru použitím symbolu „vlnovka“, tj. ~. Například: „V díle J.K. Tyla je...“ píšeme jako `V~díle J.K.~Tyla je`.

Pevná (zúžená) mezera je nezlomitelná mezera, která má pevnou délku (nesmí být rozšířena). V LaTeXu ji vytvoříme napsáním `\,` (opačné lomítko a čárka).

Zúžená mezera se používá v těchto případech:

- pro rozčlenění dlouhého čísla, například 12 004 891 (píšeme `12\,004\,891`),
- oddělení čísla a jednotky (15 kg – `15\,kg`),
- ve jméně pro oddělení iniciály od příjmení (J.K. Tyl – `J.K.\,Tyl`), pokud jsme nepoužili nezlomitelnou mezeru (je užší, takže záleží na fontu, co vypadá lépe).

V LaTeXu je skvělá možnost, jak automatizovat nahrazení běžných mezer za jednopísmennými předložkami a spojkami pevnými mezerami – existuje program *vlna* od Petra Olšáka, který automaticky přidá vlnky tam, kam budeme chtít. Pracuje na příkazovém řádku, jako parametr mu především dáme název souboru, ve kterém se má náhrada provést. Ale jako parametr můžeme dodat i seznam písmen, za kterými se má vlnka vložit:

```
vlna -v KkSsVvZzOoUuAaIi "soubor.tex"
```

Tento program se dá stáhnout ze stránky
ftp://math.feld.cvut.cz/pub/olsak/vlna/
(jsou tam především verze pro Linux, verze pro Windows je v podadresáři `oldbin`, soubor se jmenuje `vlna32.exe`).

Pokud používáme „chytrý“ editor, který je určen přímo pro LaTeX (například TeXnicCenter), je možné vytvořit si profil pro překlad, který automaticky ovlnkuje právě otevřený soubor. Například ve zmíněném programu TeXnicCenter si vytvoříme nový profil (*Build – Define Output profiles*, *Add*), necháme sice zatrhnuto *Run LaTeX in this profile*, ale jako cestu ke spustitelnému souboru uvedeme cestu k programu vlna a zadáme parametry `-v KkSsVvZzOoUuAaIi "%pm"`. Alternativně můžeme v takovémto profilu spouštět vlnu jako postprocesor.

Nově se dá použít i balíček *encxvlna*, který přímo do souboru vlnky nevloží, ale provádí to během překladu dokumentu. Stačí do preambule napsat

```
\usepackage{encxvlna}
```

(autor doporučuje dát to makro těsně před `\begin{document}`, aby byla jistota, že doopravdy budou vkládány vlnky do finálního .TEX souboru, který prošel úpravami všech vkládaných balíčků). Tento balíček se však musí předem zprovoznit, návod najdeme na
http://merlin.fit.vutbr.cz/wiki/index.php/%C4%8Cesk%C3%A1_sazba_v_LaTeXu.

Existují i další typy mezer, které lze do dokumentu vložit (v různých předdefinovaných délkách, nebo třeba pružné mezery, případně vyplněné tečkami tak, jak známe z Obsahu).

3.1.4 Speciální znaky

Někdy je třeba vkládat speciální znaky, které přímo na klávesnici nemáme. Nejběžnější jsou následující:

Trojtečka neboli *výpustka* je tvořena třemi tečkami ..., které nejsou odděleny mezerami (ve skutečnosti je mezi nimi zúžená mezera), *za trojtečku už další tečku neděláme*, i kdyby šlo o konec věty.

Trojtečku vysázíme makrem `\dots` (za ním je někdy vhodné přidat ještě prázdnou složenou závorku, abychom nezlikvidovali následnou mezeru: `\dots{}`).

Symbol krát (×) nikdy nenahrazujeme písmenem *x* („iks“)! Tento symbol najdeme v MS Wordu v dialogovém okně *Symbol*, pak píšeme třeba 3× (bez mezery), v LaTeXu použijeme matematické prostředí s makrem `$\times$` (tedy například `$3\times$`).

Promile a procento se od čísla oddělují mezerou (10 %, 2 ‰). Pokud mezeru nenapíšeme, jde o přídavné jméno „desetiprocentní“. V LaTeXu se symbol procenta používá pro zápis komentářů, takže pokud chceme, aby se ten symbol objevil ve výstupu, dáme před něj zpětné lomítko: `\%`, a v případě promile existují makra v několika balíčcích, například v balíčku *gensymb* je makro `\textperthousand`.

Stupeň se od čísla odděluje mezerou, pokud za stupněm má následovat Celsius, jinak mezeru nepíšeme (25 °C, 45°). V balíčku *gensymb* najdeme makra `\celsius` a `\degree`.

Jmenování všech možných speciálních znaků, které je možné v LaTeXu zadat, by zabralo celou knihu. Představu si můžeme udělat z různých zdrojů, které tyto symboly sdružují. Například:

- http://www.tex.ac.uk/tex-archive/info/symbols/comprehensive/symbols-a4.pdf [7],
- http://vavreckova.zam.slu.cz/didact/symbols/ (v části stránky „Ukázkové soubory s různými symboly“).

Pro zajištění určité estetické úrovně textu je důležité občas písmena mírně „posunout“, aby tvořila harmonický celek.

Slitky (ligatury) jsou shluky několika písmen, která k sobě opticky patří a lze je spojit do jednoho „multiznaku“. Typický příklad jsou dvojice písmen fi, fl, ff. MS Word se slitky sám nezabývá, naopak LaTeX je v řádkovém módu dělá automaticky. Můžeme je místně zakázat (obvykle není důvod) – srovnejte: „fi × fi“, napsáno jako `fi $\times$ f{}i`.

Podřezávání (kerning) je přisouvání znaků, aby zbytečně nevznikala příliš velká mezera mezi písmeny. Používá se u dvojic znaků, které se svými přisunutými okraji tvarově doplňují, například AV nebo Te. Ve Wordu si nanejvýš můžeme pohrát s přisouváním znaků na záložce *Proložení znaků* v dialogu *Písmo*, LaTeX tuto úpravu dělá automaticky a v OpenOffice.org se dá nastavit ve vlastnostech odstavce. Pokud

chceme tuto vlastnost potlačit, stačí mezi takovou dvojici písmen vložit prázdný blok – srovnejte: „To × To“, napsáno jako `To $\times$ T{}o`; „AV × AV“, napsáno jako `AV $\times$ A{}V`.

3.1.5 Dělení slov

V LaTeXu se dělením slov obvykle nemusíme zabývat – pokud máme správně zprovozněn jazykový balíček, slova jsou na konci řádku dělena automaticky většinou správně (ne vždy).

Dělení konkrétního slova v rámci celého dokumentu můžeme ovlivnit uvedením slova včetně jeho dělení v záhlaví v parametru makra `\hyphenation`, ale pokud jde o jedno slovo na jednom konkrétním místě dokumentu, můžeme jednoduše určit dělení takto:

```
zá\-stup\-ci progra\-mů
```

Tím, že před spojovník umístíme zpětné lomítko, dáme najevo, že na tom místě bude spojovník jen v případě, že se bude nacházet u konce řádku (pak dojde ke zlomu). Slovo obsahující dvojznak `\-` bude děleno pouze na tomto místě (a jen tehdy, když bude u konce řádku), dvojznak můžeme ve slově uvést i na více místech (pak se zvolí to místo, které je vzhledem ke konci řádku lépe umístěno).

3.2 Členění textu

3.2.1 Seznamy

V práci je možné využít běžné předdefinované seznamy itemize, enumerate a description, navíc v souboru `config.tex` jsou přidány definice seznamů s menší roztečí jednotlivých položek – *mitemize*, *menumerate* a *mdescription*, a dále seznamy s ještě menší roztečí – *citemize* a *cenumerate*. Použití je podobné, například:

```
\begin{citemize}
  \item položka1,
  \item položka2.
\end{citemize}
```

- položka1,
- položka2.

Všechny potřebné rozměry je možné ovlivnit i ručně, stačí se podívat, jak jsou uvedená prostředí vytvořena.

Za určitých okolností můžeme potřebovat dočasně číslovaný seznam „přerušit“ (například proto, že chceme vložit objekt,který si se seznamy nerozumí), ale na číslování první části chceme navázat v druhé části seznamu. Případně podobný problém řešíme tehdy, když chceme v seznamu začít číslovat od jiného čísla než 1. Postupujeme takto:

```
\begin{menumerate}
  \item První položka prvního seznamu,
  \item druhá položka prvního seznamu,
\end{menumerate}
...{} (přerušení posloupnosti)
\begin{menumerate}
\setcounter{enumi}{2}
  \item první položka druhé části,
  \item \dots
\end{menumerate}
```

1. první Položka prvního seznamu,
2. druhá položka prvního seznamu,

... (přerušení posloupnosti)

3. první položka druhé části,
4. ...

Vidíme, že stačí nastavit čítač *enumi* na hodnotu posledního čísla předchozího seznamu, resp. na číslo o 1 menší než jaké chceme u první položky následujícího seznamu. Čítač *enumi* platí pro první úroveň číslování, pro druhou je čítač *enumii*, třetí *enumiii*. To by nám mělo stačit, víc než tři úrovně může způsobit znepřehlednění seznamu.

Nyní k ukončení jednotlivých položek seznamu. Platí, že seznam je třeba koncepčně začlenit do okolního textu s tím, že záleží na rozsáhlosti textu v jednotlivých položkách.

V případě, že jsou položky krátké, jednořádkové, obvykle je seznam chápán jako součást jakési širší věty, která zřejmě začíná už před seznamem. Pak jednotlivé položky ukončujeme čárkou, na konci poslední dáme tečku (nebo čárku, pokud pomyslná věta za seznamem pokračuje). Tomu odpovídá použití velkých či malých písmen v položkách seznamu. Například:

Začátek pomyslné věty (souvětí)

Postupujeme následovně:

- otevřeme program,
- stiskneme klávesu F1,
- doufáme, že se spustí nápověda.

Konec pomyslné věty

Pokud ovšem máme v položkách poněkud více textu, chápeme jednotlivé položky jako samostatné věty, tj. začínají velkým písmenem a jsou ukončeny tečkou. V tom případě můžeme mít i více vět v jedné položce seznamu.

Před každým seznamem by měla být jakási „úvodní věta“ (alespoň pár slov), což ostatně vidíme i na ukázce výše na této straně. To je nutné zejména tehdy, když chceme seznam umístit hned pod nadpis některé (pod)kapitoly.

V souvislosti se seznamy je třeba upozornit, že bychom je neměli nadužívat. Když vidíme, že seznam zabírá například celou stránku nebo dokonce celou (pod)kapitolu, měli bychom uvažovat o změně. Můžeme například uvést zkrácenou verzi seznamu a pod tímto seznamem podrobněji rozvést jednotlivé položky v samostatných odstavcích.

3.2.2 Poznámky pod čarou

Poznámky pod čarou používáme s rozvahou. Neměli bychom zapomínat, že kdykoliv použijeme poznámku pod čarou, nutíme čtenáře, aby „klesl pohledem“ na konec stránky a po přečtení poznámky opět zdlouhavě vyhledával, kde vlastně předtím přerušil čtení. Takže můžeme je používat, ale s mírou. V LaTeXu použijeme makro

`\footnote{text poznámky pod čarou}`.

S poznámkou pod čarou bývá někdy problém, pokud ji chceme použít uvnitř některých speciálních prostředí (například potřebujeme přidat poznámku pod čarou k popisku obrázku či tabulky). Řešením je oddělení označení místa poznámky pod čarou a vkládaného textu poznámky (je to ukázáno na konkrétním příkladu, viz str. 37).

Existují také poznámky na okraj (marginal paragraphs), pro ně lze použít makro

`\marginpar{poznámka na okraj}`

Ovšem v závěrečné práci poznámky na okraj moc nevyužijeme, není důvod je používat.

3.2.3 Vícesloupcová sazba

Také v odborném textu je někdy potřeba vhodně zarovnat objekty či odstavce textu vedle sebe do více sloupců, případně uvést dlouhý seznam s krátkými položkami (což by zřejmě vytvořilo nevzhlednou dlouhou „nudli“).

Zarovnání lze řešit tabulátory, tedy prostředím *tabbing*, ale není to jediná možnost. Další možností je využití prostředí *multicols* z balíčku *multicol*. Toto prostředí používáme k vytvoření sloupců o stejné šířce.

- první,
- druhý,
- třetí,
- čtvrtý,
- pátý,
- šestý,
- sedmý,
- osmý,
- devátý,
- desátý,
- jedenáctý.

Uvedený seznam byl vytvořen následujícím kódem:

```
\begin{multicols}{4}
\begin{citemize}
\item první,
\item druhý,
\item třetí,
...
\item jedenáctý.
\item[]\relax
\end{citemize}
\end{multicols}
```

Všimněte si, že poslední (dvanáctý) prvek negeneruje žádnou položku a nemá se objevit ani odrážka. Prázdné hranaté závorky určují, že se nemá zobrazit žádný symbol odrážky, a místo textu je prázdný příkaz (`\relax`). Účelem je při menším počtu řádků v posledním sloupci zachovat správné rozteče mezi položkami (tj. vytváříme virtuální odrážky, které zajišťují pravidelnost vertikálního odsazení).

Pokud potřebujeme různou šířku sloupců, musíme použít jiný postup. Jednou z možností je použití tabulek (ovšem musíme zajistit odstranění odsazování vnitřku buněk od jejich okrajů – viz sekci o tabulkách dále), další možnost je využití *parboxů* a prostředí *minipage*.

Obecně se hodně špatně kombinují tabulky a parboxy (pokud je tabulka uvnitř a parbox vně), takže pokud chceme zarovnávat vedle sebe několik objektů, z nichž některý je tabulka, volíme i pro samotné zarovnání tabulku. Její záhlaví by mohlo vypadat například takto:

```
\begin{tabular}{@{}ccc@{}}
první objekt
&
druhý objekt
&
třetí objekt
\end{tabular}
```

Tím jsme umístili tři objekty (potenciálně různě široké, třeba obrázky, tabulky, parboxy) vedle sebe – za předpokladu, že součet jejich šířek plus rezerva nepřekračuje šířku stránky. Šířka sloupců se přizpůsobí šířce těchto objektů. Nevýhodou je, že mezery mezi objekty nejsou pružné.

3.2.4 Řízení zaplnění konkrétních stránek

Občas nám nevyhovuje, kde přesně je stránka lámána (tj. kde v posloupnosti textu končí jedna stránka a začíná následující). Délku zaplněné části jedné (konkrétní) stránky můžeme ovlivňovat makrem

`\enlargethispage{...}`

kde do parametru doplníme délku, o kterou chceme stránku prodloužit (kladný údaj) či zkrátit (záporný údaj). K číslu musíme vždy přidat i jednotku (cm, mm, em, ex, pt, apod.). Například:

`\enlargethispage{1cm}` ..... prodloužíme stránku o 1 cm (pozor, to je dost hodně)
`\enlargethispage{-2mm}` ..... zkrátíme stránku o 2 mm
`\enlargethispage{.5cm}` ..... prodloužíme stránku o půl centimetru
`\enlargethispage{1ex}` ..... prodloužíme o jeden půlčtverčík (výška písmene „x“)
`\enlargethispage{.5em}` ..... prodloužíme o půl čtverčíku (čtverčík je výška „M“)
`\enlargethispage{-4pt}` ..... zkrátíme o 4 body

Výhodou použití měr „em“ a „ex“ je jejich návaznost na základní velikost písma (jde opravdu o výšku písmen „M“ a „x“ v aktuálním fontu).

Je důležité vědět, kam toto makro umístíme, aby platilo právě pro tu stránku, kterou chceme. Většinou makro dáme na konec některého z odstavců, které určitě končí na dotyčné stránce.

Při výpočtu místa zlomu stránky nemusí konec textu vyjít vždy na tomtéž místě. LaTeX v tom případě postupuje jedním ze dvou způsobů – buď nechá spodní okraj textu napevno tam, kde vyšel, a nebo mírně zvětší všechny vertikální mezery na stránce (obvykle mezi řádky) tak, aby spodní okraj textu byl na stejném místě jako na ostatních stránkách. To můžeme ovlivnit:

`\raggedbottom` ..... nepravidelný spodní okraj, mezery se nemění
`\flushbottom` ..... zajišťuje zarovnání spodního okraje stránky

Na zalomení stránky má také vliv nastavení voleb `\clubpenalty` a `\widowpenalty`, kterých bychom si měli všímat, když nechceme, aby po stránkovém zlomu nezůstal na začátku nebo konci stránky jediný řádek odstavce (zbytek na jiné straně).

Na některých místech je třeba si vynutit zalomení stránky (přechod na novou stránku). To lze několika způsoby:

- pomocí `\newpage` přejdeme na novou stránku přesně na daném místě, vynechaná část strany zůstane nevyužita,
- makrem `\pagebreak` při současném nastavení (raggedbottom) to znamená prakticky totéž, jinak narozdíl od předchozího zajišťuje zarovnání spodního okraje stránky,
- makro `\clearpage` opět zde funguje podobně, navíc vynucuje umístění všech plovoucích prostředí (obrázků a tabulek), které dosud LaTeX nedokázal umístit, má také oboustrannou variantu `\cleardoublepage`, kterou zde však nevyužijeme,
- makro `\pagebreak` můžeme opatřit nepovinným parametrem určujícím pravděpodobnost zlomu: `\pagebreak[3]`. Ke zlomu stránky na daném místě dojde pravděpodobněji tehdy, když platí tyto podmínky:
  - je uvedené číslo větší (číslo 4 odpovídá použití makra bez parametru),
  - jsme blíže konci stránky.

Nepovinný parametr u makra `\pagebreak` používáme hlavně tehdy, když za určitým textem či objektem nechceme nechávat jen jeden či dva řádky a chceme zlom provést pouze tehdy, když je dané místo blízko konci stránky. Typická používaná hodnota je právě 3.

Také pro předčasné ukončení řádku uvnitř odstavce máme k dispozici makra `\newline` (obdoba primitiva `\\`) a `\linebreak`, u druhého můžeme přidat nepovinný parametr určující pravděpodobnost zlomu řádku.

3.2.5 Horizontální a vertikální odsazení

Horizontální (vodorovné) odsazení se většinou moc neřeší, od toho máme tabulátory, tabulky, boxy a další možnosti. Nicméně někdy se může hodit. Použijeme buď makro

`\hspace{...}` nebo primitivum `\hskip ...`, kde doplníme šířku potřebného odsazení. Pokud jako parametr napíšeme `\fill`, vytvoříme pružnou mezeru. Pro ten případ

existuje i kratší varianta – `\hfill`. Jestliže chceme pružnou mezeru vyplnit tečkami, napíšeme `\dotfill`.

Někdy je třeba upravit vertikální (svislé) odsazení, například tehdy, když do textu vkládáme některý objekt (obrázek, graf, tabulku) a standardně generované mezery opticky neladí. Pak použijeme makro `\vspace{...}` nebo primitivum `\vskip ...` Jako parametr opět použijeme délku včetně jednotky. Pokud bude parametrem záporné číslo, následná vertikální mezera se zkrátí, tedy to, co za mezerou následuje, se přiblíží. O pružných mezerách platí totéž co u horizontálních, zkrácená varianta je `\vfill`.

Varianty s hvězdičkou fungují i tehdy, když před dotyčným makrem nic není, takže pokud například nemá být nic na začátku řádku, ale chceme něco posunout až na konec řádku (zarovnat k pravému okraji), napíšeme `\hspace*{\fill}...`

U mezer, kde udáváme velikost, tedy máme na výběr mezi *primitivy* (`\hskip` a `\vskip`) a *makry* (`\hspace` a `\vspace`). Rozdíl je nejen v tom, jak zadáváme příslušnou velikost (u primitiva hned za ním, kdežto u makra ve složených závorkách), je rozdíl taky v tom, že v určitých situacích je použitelné jen jedno z toho (buď primitivum nebo makro). Jednoduše – když nefunguje jedno, použijeme druhé.

Pro vertikální i horizontání odsazení se někdy využívá obdélník s jedním rozměrem nulovým, například `\rule{10em}{0pt}` vygeneruje neviditelný obdélník o šířce 10 em a nulové výšce. Toho můžeme využít například k nenápadnému stanovení šířky sloupců tabulky (pokud nám nevadí jeden řádek navíc):

```
\begin{tabular}.....
.....
\\
\multicolumn{1}{c}{\rule{10em}{0pt}}&\multicolumn{1}{c}{\rule{7em}{0pt}}&....
\end{tabular}
```

Pružné mezery vytváříme pomocí maker `\hfill` (horizontální), `\vfill` (vertikální), `\dotfill` (tečkovaná).

3.3 Obsah a jiné seznamy

3.3.1 Obsah, seznam tabulek a seznam obrázků

Tyto seznamy v žádném případě nevytváříme ručně, jsou generovány pomocí následujících maker:

`\tableofcontents` ..... obsah, na začátku dokumentu

`\listoftables` ..... seznam tabulek, na konci dokumentu

`\listoffigures` ..... seznam obrázků, na konci dokumentu

V šabloně je stanoveno, na kterém místě a jakým způsobem mají být tyto seznamy vygenerovány. Obvykle na tom není třeba nic měnit, někdy však potřebujeme přidat do obsahu vlastní řádek. To provedeme takto:

```
\addcontentsline{toc}{section}{{ Text řádku }}
```

To by platilo pro nadpis první úrovně, pro druhou či třetí úroveň bychom místo řetězce *section* napsali *subsubsection* nebo *subsubsection*. Pozor na mezery přes uzavíracími závorkami.

Pokud chceme přidat nový řádek do seznamu tabulek, místo řetězce *toc* napíšeme *lot* (zkratka z „list of tables“), pro seznam obrázků to bude *lof* (list of figures).

Pokud je ve výsledném seznamu vkládaný řádek jinde než by měl být (řádky jsou přehozeny), může pomoci vložení makra `\clearpage` před `\addcontentsline` (samozřejmě pokud na to místo patří stránkový zlom).

Mezi občasné chyby při psaní závěrečných prací patří neaktuálnost některého z výše uvedených seznamů. V LaTeXu zajistíme aktuálnost jednoduše tak, že po ukončení práce pro jistotu soubor několikrát přeložíme (první aktualizace by mohla prodloužit obsah nacházející se na začátku dokumentu, a tedy by čísla stránek neseděla, proto více než jednou).

3.3.2 Seznam literatury

Seznam literatury je jednou z nejdůležitějších součástí práce. Jeho složení ukazuje, zda autor nastudoval dostatek kvalitních a relevantních zdrojů informací. Množství položek v seznamu literatury je relativní, záleží na konkrétním tématu. Může to být třeba jen jedna strana, a nebo stránek několik. Některé zdroje jsou studentům doporučeny už při zadání tématu práce, ale student by měl v průběhu řešení tématu hledat další zdroje (podle vlastního směřování a individuálních potřeb) a seznam literatury dále doplňovat.

Co se týče kvality a relevance, vybíráme především důvěryhodné zdroje, kde lze očekávat informace bez vážnějších chyb. Pozor, Wikipedia je sice zajímavý zdroj,

ale vzhledem ke svému charakteru rozhodně nikoliv důvěryhodný – z vlastní zkušenosti mohu potvrdit, že na některých stránkách lze najít mnoho chyb. Není řečeno, že Wikipedii nesmíme používat, ale pokud je to možné (obvykle to možné je), vybíráme spíše zdroje se známým a odborně fundovaným autorem, ideálně recenzované nebo kladně komentované.

Je možné, že jsme využili zdroje typu cizojazyčný slovník, slovník cizích slov a podobně, ovšem takové zdroje neuvádíme (pokud studium těchto zdrojů není přímo součástí tématu práce).

Při výběru zdrojů bychom měli dbát na to, aby alespoň jeden (dostatečně obsáhlý) zdroj byl cizojazyčný (typicky v angličtině), což v současné době snadno dostupného internetu není problém. Alespoň jeden zdroj by také měl být tištěný (kniha, sborník, monografie, odborný časopis apod.). Upozorňuji, že odborný a populárně odborný časopis jsou dvě různé věci, které nelze zaměňovat.

Forma položek literatury (tedy citací) musí zachovávat normu ISO 690. Pokud tuto normu plně neovládáme, můžeme použít generátor citací (viz položku [1] v seznamu literatury), ale i zde je třeba dávat pozor, jaký typ zvolíme a které údaje doplníme.

V případě článků na Internetu většinou využijeme typy *Příspěvek na webu*, *Webová stránka*, *Web*. Často bývá problém některé povinné položky zjistit, tedy uvádíme vše, co se nám zjistit podaří. Nejproblematičtějším údajem bývá název webu – někdy nezbývá, než se poradit s vedoucím práce.

Podle metodického pokynu je třeba mít obsah *uspořádán podle abecedy*, bez ohledu na typ jednotlivých zdrojů (takže tištěné a elektronické bez skrupulí pomíchejte). Protože člověk je tvor omylný a abeceda je až překvapivě dlouhá, je dobré si po dokončení práce seznam literatury několikrát projít a zkontrolovat, jestli jsme něco špatně nezařadili. Po úpravách práci nejméně jednou znovu přeložíme, aby se případné změny v pořadí (a tedy i v číslovaném označení položek) promítly do odkazů v textu.

Je vhodné v textu do seznamu literatury odkazovat, zejména tehdy, když

- vkládáme *doslovnou citaci* z cizího díla, statistickou informaci, anebo přejímáme obrázek či jiný objekt,

- chceme čtenáře, který se o problematiku chce zajímat hlouběji, nasměrovat na další informace (například formulací „další informace o této problematice nalezneme v“, „podrobněji na“), čímž dokazujeme svůj přehled v dané oblasti, popř. potřebujeme čtenáři umožnit ověření uváděné (třeba statistické) informace,
- *parafrázujeme*, což je nejběžnější případ, kdy do seznamu literatury odkazujeme.

V případě *doslovné citace* je nutné přidat odkaz do seznamu literatury, na text navíc použijeme kurzívu a uzavřeme do uvozovek (za nimi bude odkaz na zdroj). Pokud jde o knihu, měli bychom přidat i číslo strany.

Zaměřme se nyní na *parafrázování*. Znamená to, že uvádíme myšlenku někoho jiného, ale necitujeme ji doslova, tj. vlastními slovy vysvětlujeme něco, o čem jsme si přečetli v jiném díle. Přesně to děláme ve většině teoretické části práce. Za parafrázovaný text umísťujeme odkaz do seznamu literatury (ale nemusíme použít kurzívu ani uvozovky). Typicky tento odkaz umísťujeme za větu/odstavec/několik odstavců s parafrází, a pokud v odstavci vycházíme z několika zdrojů, uvedeme jich víc za sebou.

Pozor – pokud vezmeme odstavec vytvořený někým jiným a jen zaměníme pár slov za jejich synonyma (případně vezmeme text v angličtině a jen jej přeložíme), přičemž pouze uvedeme odkaz do seznamu literatury (případně ani to), nejedná se o parafrázi, ale o plagiát!

V odborných článcích bývá dokonce zvykem, že na každou položku seznamu literatury musí být v textu alespoň jeden odkaz. V závěrečné práci to nemusíme do puntíku splnit, nicméně na většinu položek by odkaz vést měl, aby bylo zřejmé, že uváděné zdroje jsme opravdu používali.

V LaTeXu do seznamu literatury odkazujeme jednoduše makrem `\cite{...}` kde jako parametr uvedeme identifikátor zdroje, který jsme mu určili v seznamu literatury. V nepovinném parametru můžeme přidat další informaci, například pro číslo strany v knize: `\cite[str. 191]{texbooknaruby}`, vysází se [6, str. 191].

3.3.3 Seznam zkratek a glosář

Seznam zkratek můžeme formovat jako tabulku (což asi bude úpravnější) nebo seznam typu *description*. Protože tatáž zkratka může mít více různých významů (podle konkrétního oboru nebo oblasti použití), uvádíme v seznamu také význam zkratky (obvykle anglický přepis, ze kterého zkratka vznikla).

Seznam zkratek nepodceňujte. Nedávejte tam „jasné“ zkratky, které zná každý (například ČR pro Českou republiku), ale z informatických zkratek byste neměli nic vynechat. Závěrečná práce z oblasti informatiky, která nemá žádné zkratky, je podezřelá, protože informatici si přece na zkratky potrpí...

Glosář je výkladový slovník, kde ke každému pojmu je připojeno vysvětlení jeho významu. Glosáře se k závěrečným pracím většinou nepřipojují, u čtenářů se předpokládá rozhled v daném oboru a znalost používaných pojmů. Pokud přece jen v textu používáme pojmy, které nejsou v oboru běžně používány, můžeme vysvětlení těchto pojmů věnovat část některé z kapitol.

3.4 Objekty a odkazování

Každý objekt, který do dokumentu vložíme (obrázek, tabulka, graf, schéma, atd.), musí mít svůj popisek (v odborných pracích včetně závěrečných vysokoškolských prací navíc číslovaný).

Pokud je to objekt přejatý (například obrázek stažený z webu nebo tabulka, jejíž obsah jsme odněkud přepsali), je třeba na konec popisku přidat informaci o zdroji objektu. To bychom měli provést pokud možno hned při vkládání, protože zejména obrázky se dohledávají velmi špatně. Pokud se dotyčný zdroj nachází v seznamu použité literatury (velmi pravděpodobně ano), použijeme křížový odkaz do tohoto seznamu (viz kap. 3.4.3 na str. 40), v opačném případě můžeme umístit odkaz na zdroj do poznámky pod čarou.

3.4.1 Tabulky

Tabulky jsou co do úpravy vzhledu poměrně variabilní záležitost, ale zásadně bychom měli použít stejný vzhled pro všechny tabulky v celé práci. Proto bychom si měli jejich vzhled ujasnit už na začátku (při vkládání první tabulky) a svého rozhodnutí

se striktně držet. Totéž se týká určení, zda popisek tabulky bude nad nebo pod dotyčnou tabulkou.

Parametr	**Vycentrováno**	**Reálné číslo**	**Cena**
Abc	Def	1 528,31	5 810 Kč
Další řádek	tabulky	2,999 93	60 Kč

Tabulka 1: Ukázka tabulky, zarovnání čísel podle desetinné čárky

Tabulka č. 1 je vytvořena tímto kódem:

```
\begin{table}[htb]
\centering
\radkovani[1.2]
  \begin{tabular}{|l||c|r@{,}l|r|}
  \hline
    \bfseries Parametr & \bfseries Vycentrováno
      & \multicolumn{2}{c|}{\bfseries Reálné číslo}
      & \multicolumn{1}{c|}{\bfseries Cena} \\ \hline\hline
    Abc  & Def  & $1\,528$  & $31$  & $5\,810$ Kč \\ \hline
    Další řádek & tabulky & $2$  & $999\,93$ & $60$ Kč \\ \hline
  \end{tabular}
  \caption{Ukázka tabulky, zarovnání čísel podle desetinné čárky}
  \label{tab:parametry}
\end{table}
```

Parametr	**Vycentrováno**	**Reálné číslo**	**Cena**
Abc	Def	1 528,31	5 810 Kč
Další řádek	tabulky	2,999 93	60 Kč

Tabulka 2: Jiný layout

Tabulka 2 byla vytvořena tímto kódem:

```
\begin{table}[htb]
\centering
\radkovani[1.2]
  \begin{tabular}{@{}l||c|r@{,}l|r@{}}
  \hline
    \bfseries Parametr & \bfseries Vycentrováno
      & \multicolumn{2}{c|}{\bfseries Reálné číslo}
      & \multicolumn{1}{c}{\bfseries Cena} \\ \hline\hline
    Abc  & Def  & $1\,528$  & $31$  & $5\,810$ Kč \\ \hline
    Další řádek & tabulky & $2$  & $999\,93$ & $60$ Kč \\ \hline
  \end{tabular}
  \caption{Jiný layout}
  \label{tab:parametry2}
\end{table}
```

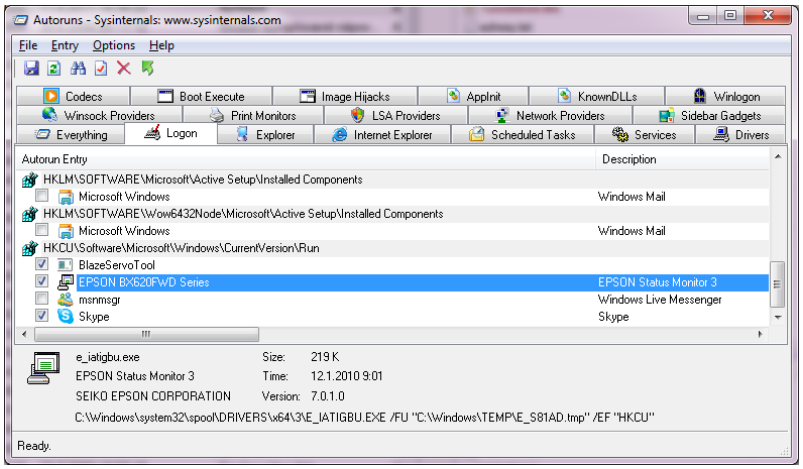
Jak vidíme, rozdíl je především ve specifikaci typu sloupců tabulky. Všimněte si, jakým způsobem je odstraněna mezera mezi vnějšími bočními okraji tabulky a obsahem přilehlých buněk – na začátku a na konci specifikace sloupců máme přidáno `@{}`, což právě způsobí odstranění „nadbytečných“ mezer.

V tabulkách 1 a 2 je ukázka pouze základního formátování tabulky. Kromě toho, že můžeme hlouběji ovlivňovat zarovnání v buňkách, slučovat sloupce či řádky apod., parametry tabulky (jako odsazení obsahu buňky od ohraničení) se dají ovlivnit makry `\renewcommand{\arraystretch}{1.2}` a `\tabcolsep=2pt` (konkrétní hodnoty dosadíme podle potřeby).

3.4.2 Obrázky

Obrázky v odborném textu mají vhodně doprovázet a metodicky dotvářet dokument. U některých témat nejsou zapotřebí, u jiných jsou oproti tomu naprosto nutné, zejména tam, kde obrázek s nákresem, grafem či diagramem usnadňuje pochopení popisovaného postupu. Obrázky vkládáme buď přímo do textu (jak je to i v tomto dokumentu), nebo do příloh (v případě, že jejich logická vazba na text je spíše menší a nebo jsou velmi rozsáhlé, případně je chceme tisknout na jiné (barevné, kvalitnější) tiskárně nebo na jiný papír).

Na obrázky se můžeme odkazovat makrem `\ref`. Například na obrázek, u jehož popisku je dopsáno `\label{fig:autoruns}` (což je právě níže zobrazený obrázek), odkazujeme `na obrázku \ref{fig:autoruns}`, načež se zobrazí „na obrázku 1“.



Obrázek 1: Ukázka vložení obrázku – program Autoruns od Sysinternals

Protože zásadně necvakáme všechny soubory na jednu hromadu, ale umísťujeme je do struktury složek, je třeba uvést cestu včetně umístění. Názvy složek na cestě oddělujeme lomítkem (obyčejným, ne opačným). Uvedený obrázek byl vložen následujícím kódem:

```
\begin{figure}[htb]
\centering
   \includegraphics[width=.7\textwidth]{pictures/autoruns}
  \caption{Ukázka vložení obrázku - program Autoruns od Sysinternals}
  \label{fig:autoruns}
\end{figure}
```

Určitě jste si všimli, že v názvu vkládaného souboru není uvedena jeho přípona. Pokud máme na daném umístění tentýž obrázek ve více různých formátech s různými příponami, LaTeX si vybere tu variantu, která je pro daný způsob překladu nejvhodnější. Ale pokud chceme použít soubor s konkrétní příponou, není problém ji uvést.

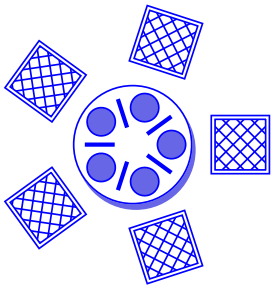
Pokud vkládáme obrázky, jejichž autorem je někdo jiný, podle autorského zákona *musíme přidat informaci o zdroji* obrázku – odkazem do seznamu literatury (v případě, že tam dotyčný zdroj máme) nebo v poznámce pod čarou (když není v seznamu literatury). Ovšem poznámka pod čarou může v popisku obrázku dělat problémy, které vyřešíme vhodným umístěním maker `\footnotemark` (vygeneruje označení poznámky pod čarou na místě) a `\footnotetext` (zadáme text poznámky) takto:

```
...
\caption[Popisek obrázku]{Popisek obrázku\footnotemark}
...
\end{figure}%
\footnotetext{Text poznámky pod čarou, třeba Zdroj: ...}
```

Jestliže máme v textu velmi mnoho menších obrázků, je neefektivní nechávat na stranách kolem nich zbytečně mnoho vynechaného místa. Existuje více řešení:

1. Menší obrázky můžeme sdružit do jednoho prostředí figure, pokud k sobě logicky patří, a samozřejmě to musíme také zohlednit v popisku obrázku.
2. Použijeme prostředí `wrapfigure`, k tomu je třeba načíst balíček *wrapfig* (v záhlaví dokumentu uvedeme `\usepackage{wrapfig}`), ukázku včetně komentáře vidíme níže. V balíčku *wrapfig* je definováno také prostředí pro obtékané tabulky – *wraptable*.

3. Použijeme makro `\parpic` definované v balíčku *picins*. Oproti *wrapfigure* je méně problémů s okolním textem, používáme je hlavně tehdy, když nepotřebujeme vytvoření popisku a automatické číslování. Ukázku taktéž vidíme níže.
4. Můžeme využít vícesloupcovou sazbu. Nejlepší a nejpružnější možností pro vícesloupcovou sazbu je prostředí *multicols* z balíčku *multicol*.
5. Text a objekty na stránce se dají poskládat pomocí maker `\parbox`. Méně znalý může mít problém se zarovnáním na účaří.
6. Poslední možností je využití tabulek.



Obrázek 2: Obrázek obtékaný textem

Prostředí *wrapfigure* zajistí obtékání obrázku. Struktura prostředí je podobná prostředí *figure*, jen zadáváme trochu jiné parametry: `[9]{r}{.24\textwidth}`

- počet řádků určujících výšku objektu,
- zarovnání r (doprava), l (doleva),
- šířka objektu, obvykle stejná jako šířka obrázku.

Toto prostředí nelze vložit do některých jiných prostředí, v případě špatného zobrazení musíme prostě umístit jinam. Další oblastí, kde se vyskytují problémy, je případ, kdy v obtékaném textu je některý typ seznamu. V takovém případě často stačí pohrát si s počtem řádků v parametrech (v seznamu je každá odrážka brána jako jeden řádek, i když zabírá řádků více).

Objekt, který obsahuje obrázek č. 2, je představován tímto kódem:

```
\begin{wrapfigure}[9]{r}{.24\textwidth}
  \includegraphics[width=.23\textwidth]{pictures/pethladovych}
  \caption{Obrázek obtékaný textem}
  \label{fig:pethladovych}
\end{wrapfigure}
Prostředí \emph{wrapfigure} ...
```

Jednoduché malé objekty, které nevyžadují vložení titulku (například logo produktu), můžeme vkládat pomocí makra `\parpic`. Toto makro má hodně nepovinných parametrů, často si vystačíme i se základním nastavením (objektem může být například obrázek vkládaný makrem `\includegraphics`):

```
\parpic{objekt}
\noindent
Následující odstavec, který bude objekt obtékat. Objekt je u~levého okraje.
```

Syntaxe celého příkazu je následující:

```
\parpic(šířka,výška)(posunX,posunY)[volby][pozice]{objekt}
Následující odstavec, který bude objekt obtékat.
```

Parametry v kulatých závorkách mají stejný význam jako u prostředí *picture*, jen píšeme i jednotku délky (cm, pt, atd.) a posun funguje trochu zvláštně (použijte metodu pokus–omyl). Volby určují, kam má být objekt zarovnán, také je možné zajistit orámování (jednoduché, oválné, 3D, apod.). Pozice stanovuje, jak má být objek zarovnán uvnitř vytvářeného boxu. Obrázek na začátku tohoto odstavce (logo programu CPU-Z) byl vložen takto:

```
\parpic(2.2em,2.2em)(0pt,2.8em){
  \includegraphics[height=2.2em]{pictures/logo_cpuz}}
\noindent
Parametry ...
```

Dokumentace makra `\parpic` a také dalších maker z balíčku *picins* je v němčině, ale je možné získat také stručný popis (celkem dostačující) v angličtině v archivu CTAN na adrese http://www.ctan.org/pkg/picins.

V případě, že chceme umístit více menších obrázků či obecně objektů do společného prostředí *figure*, můžeme chtít zajistit jejich označení v rámci této skupiny, případně ke každému přidat vlastní popisek. K tomu slouží balíček *subfigure* s makrem `\subfigure`. Ukázku vidíme na obrázku 3, kde je možné se automaticky odkazovat i na jednotlivé vložené subjekty jako je obrázek 3(a).

(a) První dílčí popisek

(b) Druhý dílčí popisek

(c) Třetí dílčí popisek

Obrázek 3: Hlavní popisek

Kód obrázku 3:

```
\begin{figure}[htb]
\centering
\newcommand{\pozmenit}[2][15]{\rotatebox{#1}{\small\color{modra}#2}}

\subfigure[První dílčí popisek]{
  \pozmenit{\parbox[3em]{10em}{První\centering\\[-8pt]vložený objekt}}
  \label{subfig:prvnidilci}}
\subfigure[Druhý dílčí popisek]{
  \framebox[10em]{\pozmenit{\parbox[3em]{10em}{Druhý\centering\\[-8pt]
  vložený objekt}}}}

\subfigure[Třetí dílčí popisek]{
  \pozmenit[-12]{\parbox[3em]{10em}{Třetí\centering\\[-8pt]vložený objekt}}}
\subfigure{
  \pozmenit[-12]{\parbox[3em]{10em}{Vložený objekt\centering\\[-8pt]bez
  vlastního popisku}}}
\caption{Hlavní popisek}
\label{fig:prikladsubfigure}
\end{figure}
```

3.4.3 Odkazy, reference

Jak bylo výše naznačeno, u všech přejatých objektů je třeba uvést odkaz na původní zdroj. V žádném případě tyto odkazy neklepeme ručně, tím si můžeme přivodit značné problémy při aktualizaci souboru! Pro přehlednost si shrneme nejčastější způsoby odkazování v tabulce 3 na straně 40.

Typ odkazování	**Jak je označen cíl**	**Jak na cíl odkazujeme**
Kapitola, sekce	`\label{chap:navesti}`	`\ref{chap:navesti}`
Číslo stránky	`\label{chap:navesti}`	`\pageref{chap:navesti}`
Obrázek, nákres, diagram	`\label{fig:obrazek}` (za `\caption`)	`\ref{fig:obrazek}`
Tabulka	`\label{tab:tabulka}` (za `\caption`)	`\ref{tab:tabulka}`
Položka seznamu literatury	`\bibitem{polozkacitace}`	`\cite{polozkacitace}`

Tabulka 3: Typy odkazů a referencí

Konkrétní řetězce návěští si volíme sami, ale pro přehlednost je vhodné zachovávat konvence použití chap:, fig:, tab: a případně dalších příznaků coby začátku návěští (abychom na první pohled poznali, na co odkazujeme). Pro přehlednost dále uvedeme několik příkladů.

Tato podkapitola má název doplněný návěštím:

`\subsubsection{Odkazy, reference}\label{chap:odkazyref}`, a tedy je možné na ni od-

kazovat následovně: „V kapitole 3.4.3 na straně 40 najdeme…“ – což jsme zapsali takto:

```
V kapitole \ref{chap:odkazyref} na straně \pageref{chap:odkazyref} najdeme\dots
```

Výše uvedená tabulka má také své návěští:

```
...
\end{tabular}
\caption{Typy odkazů a~referencí}
\label{tab:typyodkazu}
\end{table}
```

Na tuto tabulku se proto budeme odkazovat následovně: „V tabulce 3 na straně 40 jsou uvedeny…“, což zapisujeme takto:

```
V~tabulce \ref{tab:typyodkazu} na straně \pageref{tab:typyodkazu} jsou uvedeny\dots
```

Co se citací týče, podrobnější informace nalezneme ve zdroji [4], včetně příkladů. Kromě jiného se tam dočteme: „*Řazení citací v této příloze neodpovídá řazení dle normy; v závěrečné práci mají být všechny citace řazeny abecedně dle prvního autora bez ohledu na jejich druh.*“[4, str. 8]

Předchozí odstavec jsme vytvořili takto (zkráceno):

```
Co se citací týče,...\cite{metodickypokyn}, včetně příkladů. Kromě jiného...
dočteme: ...\uv{\emph{Řazení citací...jejich druh.}}\cite[str. 8]{metodickypokyn}
```

4 Matematická sazba a další odborné prvky

Matematická sazba zahrnuje zápis matematických, fyzikálních a chemických vzorců, ale také například formátování definic a vět. Způsobu zápisu definic, vět a důkazů se věnujeme v kapitole 4.5, zde se zaměříme na samotná matematická prostředí a zápis vzorců.

Není v možnostech tohoto textu popisovat vše, co LaTeX umí v matematické sazbě a jak se co dělá. Zaměříme se hlavně na pravidla, která je třeba dodržovat v odborném textu.

4.1 Sazba jednořádkových vzorců

Kratší vzorce lze vnořit přímo do textu, delší vzorce však zásadně z textu vydělujeme do speciálního matematického prostředí (třeba *equation*) – nazýváme je „vysazené vzorce“, a pokud je to třeba, zajistíme číselné označení. Přímo v textu můžeme sázet například takto: $x = \sum_{i=1}^{n} p_i$, což jsme získali zápisem

```
$x=\sum_{i=1}^{n}p_i$
```

Prostředí *equation* se používá následovně:

```
\begin{equation}
\label{eq:ukazkaequation}
  x=\sum_{i=1}^{n}p_i
\end{equation}
```

$$x = \sum_{i=1}^{n} p_i \tag{1}$$

Na uvedený vzorec se můžeme odkazovat běžným způsobem – vzorec (1), vy-

generováno kódem `(\ref{eq:ukazkaequation})`, kulaté závorky dodáváme ručně. Pokud vzorce nechceme číslovat, použijeme místo prostředí *equation* zdvojené symboly dolaru – `$$vzorec$$`.

Všimněte si rozdílu ve vzhledu indexů sumy v číslovaném matematickém prostředí – v textovém vzorci jsou oba indexy zarovnány tak, aby nenarušovaly výšku řádku, naopak ve vysazeném prostředí jsou umístěny přehledněji.

Vysazené vzorce se implicitně centrují. Pokud chceme, aby se zarovnávaly k levému okraji, přidáme do voleb třídy dokumentu (k makru `\documentclass` do hranatých závorek) volbu `fleqn`. Odsazení vzorce od levého okraje pak řídíme nastavením délky

`\setlength{\mathindent}{...}`, obvyklá hodnota je `0pt`.

4.2 Základní pravidla pro matematickou sazbu

4.2.1 Jak co formátovat

Proměnné sázíme matematickou kurzívou, zatímco *názvy funkcí* vzpřímeným patkovým písmem (tj. antikvou). V matematickém prostředí tedy proměnné píšeme volně bez nutnosti dalšího formátování, zatímco u funkcí použijeme příslušné makro (obvykle `\mathrm{...}`) a nebo řetězec uzavřeme do makra `\mbox{...}`, případně použijeme speciální makro generující přímo daný název funkce.

Kdy kterou možnost použít?

- Pokud existuje makro generující název funkce, použijeme toto makro. Seznam najdeme v každé učebnici či manuálu TEXu či LATEXu. V předchozím textu jsme viděli takto použité makro `\sum`. Existují makra pro všechny nejběžnější funkce jako jsou goniometrické funkce, logaritmy, odmocnina, minimum, maximum, apod.
- Makro `\mathrm{...}` je nejběžnější pro ty funkce, pro které nemáme speciální makro. Jednoduše tak vysázíme text antikvou.
- Makro `\mbox{...}` používáme spíše tehdy, když chceme do vzorce vepsat běžný text (je vidět v ukázce prostředí *eqnarray* na straně 48). Ve skutečnosti nenastavuje matematickou antikvu, ale vytvoří v prostředí nezávislý box s fontem z běžného textu. Výhodou je, že správně interpretuje textové mezery a diakritiku, takže v něm lze použít češtinu.

Další pravidla matematické sazby:

- Vektory sázíme tučnou kurzívou, např. $\mathbf{v}, \mathbf{r}$ (`\mathbf{v}`), případně nad nimi může být šipka. Ale pozor, vektory coby proměnné se sázejí (obyčejnou) kurzívou.
- Tenzory a názvy matic sázíme bezpatkovým fontem, např. M (`\mathsf{M}`).
- Konstanty a označení různých speciálních čísel sázíme antikvou, aby se nezaměňovaly s proměnnými, např. d (`\mathrm{d}`, znak diferenciálu). Pro některé konstanty existuje i speciální makro, například π (`\pi`).
- Veličiny, které se skládají z čísla a jednotky, sázíme s menší než běžnou mezerou a jednotka má být antikvou, např. 10 km (`10\,\mathrm{km}`). Toleruje se použití nezlomitelné mezery místo zúžené (tzv. vlnka), tedy `10~\mathrm{km}`.

Častou chybou je právě porušení pravidla u diferenciálu (třetí odrážka), který se používá při sázení vzorců s integrály.

Toto pravidlo se nutně uplatňuje také při sázení chemických vzorců (míněny vzorce chemických sloučenin), protože značky chemických prvků jsou vlastně obdobou konstant. Proto se většinou celé chemické vzorce vloží do makra `\mathrm`, například H_2SO_4 vygenerujeme `\mathrm{H_2SO_4}`.

4.2.2 Písmo, mezery a speciální znaky

Konvence pro fonty v matematickém režimu jsou uvedeny v předchozím textu. Podíváme se na další konvence, které souvisejí se závěrečnými pracemi na vysokých školách.

Množinu přirozených, reálných apod. čísel (obecně jakoukoliv „konstantní“ množinu se speciálním označením) obvykle sázíme buď vzpřímeným tučným písmem (přinejhorším, když nic jiného neumíme) nebo lépe matematickým kaligrafickým písmem a nebo ještě lépe zdvojenými písmeny (na to však potřebujeme některý z matematických balíčků (třeba *amsfonts* nebo *amssymb*, příp. *pxfonts*). Pro množinu reálných čísel se někdy používá speciální primitivum `\Re` generující znak $\Re$. Ale ve skutečnosti jde o znázornění reálné části komplexního čísla (imaginární část by byla $\Im$, tedy `\Im`).

Popis	**Zápis**	**Výsledek**
Tučné písmo (spíše nepoužívat)	`\mathbf{R},\mathbf{N},\mathbf{C}`	$\mathbf{R}, \mathbf{N}, \mathbf{C}$
Kaligrafické písmo	`\mathcal{R},\mathcal{N},\mathcal{C}`	$\mathcal{R}, \mathcal{N}, \mathcal{C}$
Zdvojené vzpřímené písmo (balíček *amsfonts*)	`\mathbb{R},\mathbb{N},\mathbb{C}`	$\mathbb{R}, \mathbb{N}, \mathbb{C}$

Tabulka 4: Sazba symbolů množin reálných, přirozených a celých čísel

Je třeba si uvědomit, že sazba mezer probíhá v matematickém režimu úplně jinak než v režimu textovém. Konkrétně – mezery v kódu se ignorují, naopak LaTeX mezeruje sám dle svého uvážení. Pokud nám toto uvážení nevyhovuje, můžeme je mírně upravit, což v závěrečných pracích někdy děláme, aby vzorce vypadaly přehledně a esteticky vyváženě. Konkrétní informace najdeme v každém slušnějším manuálu (či učebnici, příručce) systému LaTeX. Obvykle si vystačíme s mezerami `\,` `\:` `\;` `\ ` `\!` (po řadě od nejmenší, poslední s vykřičníkem je záporná mezera).

Malá řecká písmena se obvykle používají coby proměnné představující řetězce (např. α, β, γ – `\alpha,\beta,\gamma`) či funkce (δ – `\delta`) nebo konstanty, jedna konstanta se využívá pro reprezentaci prázdného řetězce (ε – `\varepsilon`, v zahraniční literatuře to může být λ – `\lambda`), další představuje Ludolfovo číslo (π – `\pi`). Velká (většinou vzpřímená) řecká písmena většinou značí množiny (Γ – `\Gamma`) a nebo jde o speciální operátory (Σ – `\Sigma`). Symbol Δ (`\Delta`) značí rozdíl.

V matematických důkazech se můžeme setkat se symbolem $\aleph$ (`\aleph`). Například $\aleph_0$ je mocnost (počet prvků) množiny přirozených čísel, tedy $|\mathbb{N}| = \aleph_0$.

Co se týče dalších užitečných symbolů (nejen) pro matematickou sazbu, mohu odkázat na zdroj [7] ze seznamu literatury, kde je velmi obsáhlý (ale nikoliv úplný) seznam nejrůznějších symbolů včetně jejich vyobrazení a balíčků, ve kterých jsou symboly definovány.

V oblasti matematické logiky a obecně při dokazování můžeme potřebovat následující symboly:

- $\vee$ `\vee`
- $\wedge$ `\wedge`
- $\Rightarrow$ `\Rightarrow`
- $\Leftarrow$ `\Leftarrow`
- $\Leftrightarrow$ `\Leftrightarrow`
- $\Longrightarrow$ `\Longrightarrow`
- $\Longleftarrow$ `\Longleftarrow`
- $\Longleftrightarrow$ `\Longleftrightarrow`
- $\forall x$ `\forall\,x`
- $\exists x$ `\exists\,x`
- $\bigvee_{i=0}^{n}(r \to p_i)$ `\bigvee_{i=0}^n (r\to p_i)`
- $\bigwedge_{i=0}^{n}(r \to p_i)$ `\bigwedge_{i=0}^n (r\to p_i)`

Pro zřetězení můžeme použít buď vystředěnou tečku (`\cdot`), nebo kroužek $\circ$ (`\circ`), pro násobení je buď `\cdot` nebo symbol $\times$ (`\times`). V některých důkazech využijeme trojúhelníky $\triangleleft$ a $\triangleright$ (`\triangleleft` a `\triangleright`).

4.3 Řízení umístění a vzhledu jednotlivých prvků sazby

4.3.1 Umístění objektů nad sebe, zlomky

Ve vzorcích často potřebujeme umístit jednotlivé prvky vertikálně nad sebe. Pokud se jedná o dva prvky (jeden nad druhým), je situace velmi jednoduchá. Máme tyto možnosti:

- `\frac{x}{y}` generuje zlomek: $\frac{x}{y}$, délka zlomkové čáry se přizpůsobuje, a při vnořování se přizpůsobuje i velikost písma:

```
$$
\frac{\frac{3x^2}{x^2-1}-1}{1+\frac{x}{x-1}}
$$
```

$$\frac{\frac{3x^2}{x^2-1}-1}{1+\frac{x}{x-1}}$$

- `{x \atop y}` má podobný výsledek, ale nevytvoří se zlomková čára: $\begin{matrix} x \\ y \end{matrix}$. Podobně jako předchozí, i zde se přizpůsobuje velikost písma.

- `{x \choose y}` funguje podobně, ale navíc celou strukturu uzavře do kulatých závorek: $\binom{x}{y}$. Používáme pro zápis binomických koeficientů.

- `\stackrel{x}{y}` zmenší první argument a umístí ho nad druhý, například `\stackrel{\alpha}{\longrightarrow}` vygeneruje $\stackrel{\alpha}{\longrightarrow}$

- Další možností je využití prostředí *array* vnořeného do matematického prostředí. Narozdíl od předchozích se sice nepřizpůsobuje velikost písma, ale na druhou stranu lze skládat jakékoliv množství objektů vertikálně i horizontálně a vkládat čáry stejně jako v prostředí *tabular*.

```
$$
A_{m,n}=\left[\begin{array}{cccc}
a_{1,1}&a_{1,2}&\cdots&a_{1,n}\\
a_{2,1}&a_{2,2}&\cdots&a_{2,n}\\
\vdots&\vdots&\ddots&\vdots\\
a_{m,1}&a_{m,2}&\cdots&a_{m,n}
\end{array}\right]
$$
```

$$A_{m,n}=\left[\begin{array}{cccc}
a_{1,1}&a_{1,2}&\cdots&a_{1,n}\\
a_{2,1}&a_{2,2}&\cdots&a_{2,n}\\
\vdots&\vdots&\ddots&\vdots\\
a_{m,1}&a_{m,2}&\cdots&a_{m,n}
\end{array}\right]$$

Někdy potřebujeme jednoduše umístit čáru nad nebo pod některým objektem. K tomu slouží makra `\overline{...}` (čára nad objektem) a `\underline{...}` (čára pod objektem). Tato makra lze i vnořovat. Pomocí makra `\overline{...}` můžeme zapsat třeba známé *DeMorganovo pravidlo*: $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$, což jsme vytvořili pomocí kódu

```
L_1\cap L_2 = \overline{\overline{L_1}\cup\overline{L_2}}
```

Může existovat požadavek na dvojité podtržení (třeba výsledku výpočtu) – to lze provést dvěma různými způsoby, každý je vhodný v určité situaci.

- Většinou stačí vnořit výraz do zdvojeného `\underline{...}`: $\underline{\underline{x = 25}}$ je vygenerováno kódem `\underline{x=25}`.

- Pokud je celý výpočet zarovnáván pomocí prostředí *array* nebo *eqnarray* (například aby v rovnicích byla rovnítka vždy pod sebou), bude zřejmě i výsledek

ve více sloupcích. Pak využijeme `\cline{od-do}` nebo `\hline` pro vytvoření čáry přes několik sloupců (dvakrát za sebou, když chceme dvě čáry). Jestliže na bocích čára přečnívá (horní řádky jsou širší), je třeba vytvořit víc sloupců a širší buňky uzavřít do `\multicolumn{sloupců}{zarov}{...}`. Zde se však můžeme setkat s problémem v případě, že první („virtuální“) sloupec nebude vůbec použit (pak by měl nulovou šířku). Na řádku s výsledkem bychom pak měli tento prázdný sloupec zaplnit alespoň obdélníkem s nulovou výškou (šířku doplníme podle potřeby):

```
\rule{...}{0pt} & x & = & 25 & \\ \cline{2-4}\\[-10pt]\cline{2-4}
```

Nad objektem nemusí být zrovna čára, může tam být šipka symbolizující vektor. Kód

```
\vec{\mathbf{u}}\cdot\vec{\mathbf{v}}=u_1v_1+u_2v_2=
  |\vec{\mathbf{u}}|\cdot|\vec{\mathbf{v}}|\cdot \cos \alpha
```

generuje následující: $\vec{\mathbf{u}} \cdot \vec{\mathbf{v}} = u_1v_1 + u_2v_2 = |\vec{\mathbf{u}}| \cdot |\vec{\mathbf{v}}| \cdot \cos\alpha$

Obdobou šipky nad písmenem je vodorovná čára nad písmenem. Tu vytvo-

říme takto: `\bar{a}`, výsledek: $\bar{a}$. Zatímco `\overline` můžeme použít na jakýkoliv objekt (i řetězec o více symbolech), `\bar` se dá použít jen na jediný symbol.

Podobně se používají makra `\overbrace{...}` a `\underbrace{...}` generující složenou závorku nad nebo pod objektem, ale navíc lze pomocí exponentů ještě nad (resp. pod) tuto závorku umístit další (zmenšený) objekt. Vysázení složené závorky pod objektem opatřené titulkem vidíme na následujícím kódu (počet variací m-té třídy z n prvků).

```
$$
\begin{array}{rcl}
\mathrm{P}_n^m
  &=&\prod\limits_{i=0}^{m-1}(n-i)\\
  &=&\underbrace{n(n-1)(n-2)
    \dots(n-m+1)}
    _{\mbox{celkem $m$ činitelů}}\\
  &=&\underline{\underline{
    \begin{array}{@{}c@{}}
      n!\\ \hline (n-m)!
    \end{array}}}
\end{array}
$$
```

$$\begin{array}{rcl}
\mathrm{P}_n^m &=& \prod\limits_{i=0}^{m-1}(n-i)\\
&=& \underbrace{n(n-1)(n-2)\dots(n-m+1)}_{\text{celkem } m \text{ činitelů}}\\
&=& \underline{\underline{\dfrac{n!}{(n-m)!}}}
\end{array}$$

Na příkladu je vidět také využití makra `\underline` pro dvojité podtržení výsledku. Vzhledem k tomu, že výsledek máme uzavřen v prostředí *array*, druhou

možností (se stejným výsledkem) a navíc jednodušší by bylo vytvoření dvojité čáry přímo v prostředí *array*:

```
= \begin{array}{@{}c@{}} n!\\ \hline (n-m)!\\ \hline\hline\end{array}
```

pak by už nebylo nutné podtrhávat pomocí `\underline`. Všimněte si sekvence `@{}` v parametru prostředí *array*. Jejím účelem je odstranit nadbytečné odsazení obsahu prostředí zprava a zleva, aby zlomková čára nepřečnívala.

4.3.2 Víceřádkové vzorce

Jestliže je vzorec (případně matice či jiný objekt) složitější a je nutné ho zapsat na více řádků, nebudeme zarovnání těchto řádků „bastlit“ ručně, ale použijeme příslušné prostředí (*eqnarray*), případně prostředí *array* uvnitř prostředí *equation*.

Za běžných okolností si vystačíme s prostředím *eqnarray*, které automaticky čísluje jednotlivé řádky vzorce. Pokud některý řádek nechceme číslovat, použijeme na něm makro `\nonumber`, případně pokud nechceme číslovat žádný řádek, využijeme verzi prostředí s hvězdičkou. Následuje ukázka kódu (vzorce pro konvergentní posloupnosti), kde první řádek nemá být označen číslem, ostatní ano, na očíslované řádky se lze odkazovat:

```
\begin{eqnarray}
  \mbox{konvergentní}\ \{a_n\}_{n\in\mathbb{N}},\ \{b_n\}_{n\in\mathbb{N}}\colon
    && \nonumber\\
  \lim_{n\rightarrow\infty}(a_n \pm b_n)
    & = & \lim_{n\rightarrow\infty} a_n \pm \lim_{n\rightarrow\infty} b_n\\
  \lim_{n\rightarrow\infty}(a_nb_n)
    & = & \lim_{n\rightarrow\infty} \cdot \lim_{n\rightarrow\infty} b_n
\end{eqnarray}
```

Výsledkem je následující třířádkový vysazený vzorec:

$$\begin{aligned} &\text{konvergentní } \{a_n\}_{n\in\mathbb{N}},\ \{b_n\}_{n\in\mathbb{N}}\colon \\ &\lim_{n\to\infty}(a_n \pm b_n) &&= \lim_{n\to\infty} a_n \pm \lim_{n\to\infty} b_n && (2)\\ &\lim_{n\to\infty}(a_nb_n) &&= \lim_{n\to\infty} \cdot \lim_{n\to\infty} b_n && (3) \end{aligned}$$

Všimněte si, že číslování není od (1) – vzorec s tím číslem najdeme u prvního příkladu této kapitoly, na straně 42.

Můžeme potřebovat předsadit první řádek před ostatní a zajistit jeho nezávislost na odsazování v rámci ostatních řádků. K tomu slouží makro `\lefteqn`:

```
\begin{eqnarray*}
\lefteqn{\lim_{n\rightarrow\infty}\left(\sqrt{n^2+n}-n\right)\ =}\\
&=&\lim_{n\rightarrow\infty}
\frac{\left(\sqrt{n^2+n}-n\right)\left(\sqrt{n^2+n}+n\right)}{\sqrt{n^2+n}+n}\ =\\
&=&\lim_{n\rightarrow\infty}
  \frac{n^2+n-n^2}{\sqrt{n^2+n}+n}\ =\\
&=&\lim_{n\rightarrow\infty}
  \frac{1}{\sqrt{1+\frac{1}{n}}+1}
    \ =\ \frac{1}{1+1}\ =\ \underline{\underline{\frac{1}{2}}}
\end{eqnarray*}
```

Zvolili jsme prostředí *eqnarray** s hvězdičkou, kde žádný řádek nebude číslován. Výše uvedený kód vysází následující vzorec:

$$\begin{aligned}
\lim_{n\to\infty}\left(\sqrt{n^2+n}-n\right) &= \\
&= \lim_{n\to\infty}\frac{\left(\sqrt{n^2+n}-n\right)\left(\sqrt{n^2+n}+n\right)}{\sqrt{n^2+n}+n} = \\
&= \lim_{n\to\infty}\frac{n^2+n-n^2}{\sqrt{n^2+n}+n} = \\
&= \lim_{n\to\infty}\frac{1}{\sqrt{1+\frac{1}{n}}+1} = \frac{1}{1+1} = \underline{\underline{\frac{1}{2}}}
\end{aligned}$$

Prostředí *array* uvnitř *equation* používáme ve složitějších případech, kdy je nutné určitým způsobem zarovnat větší množství prvků. V tomto prostředí můžeme používat většinu prvků, které známe z tabulek (prostředí *tabular*), včetně kreslení čar, i dvojitých. Toto prostředí lze také rekurzívně vnořovat. Následují dvě ukázky, ve kterých bychom s prostředím eqnarray nevystačili – potřebujeme více sloupců a také jinak zarovnaných. V prvním příkladu je matice uzavřena do kulatých závorek, ale je možné použít i hranaté závorky (ovšem v celém dokumentu by to mělo být stejně).

```
$$\left(\begin{array}{@{}r}
  1\\0\\-2
\end{array}\right) \cdot
\left(\begin{array}{rrr}
  0&1&2
\end{array}\right) =
\left(\begin{array}{rrr}
  0&1&2\\ 0&0&0\\ 0&-2&-4
\end{array}\right)$$
```

$$\left(\begin{array}{r}1\\0\\-2\end{array}\right)\cdot\left(\begin{array}{rrr}0&1&2\end{array}\right)=\left(\begin{array}{rrr}0&1&2\\0&0&0\\0&-2&-4\end{array}\right)$$

$$\delta_n(r,x)=\begin{cases}\delta(r,x); & r\in \mathrm{Q}-\mathrm{F}, x\in\Sigma,\\ \delta(r,x)\cup\delta(\mathrm{q}_0,x); & r\in\mathrm{F}, x\in\Sigma,\\ \delta(\mathrm{q}_0,x); & r=\mathrm{s}_0, x\in\Sigma.\end{cases}$$

```
$$\delta_n(r,x)=\left\{
\begin{array}{l@{}l}
  \delta(r,x); 
  & r\in \mathrm{Q}-\mathrm{F},
    x\in\Sigma,\\
  \delta(r,x) \cup
    \delta(\mathrm{q}_0,x); 
  & r\in \mathrm{F},
    x\in\Sigma,\\
  \delta(\mathrm{q}_0,x); 
  & r=\mathrm{s}_0,
    x\in\Sigma.
\end{array}\right.$$
```

Jak vidíme, takto lze vytvářet i různé svorky. Kdybychom chtěli vodorovnou svorku, můžeme použít rotaci boxu.

4.3.3 Pružné závorky a operátory

V předchozích uvedených vzorcích vidíme také použití pružných závorek. Takto můžeme využívat všechny typy závorek (v příkladech vidíme kulaté závorky pro matice a levou složenou závorku pro výčet různých případů, ale platí to i pro ostré, hranaté či rovné závorky). Pružnost závorek zajistíme tak, že před levou napíšeme `\left` a před pravou `\right`. Není nutné, aby obě závorky byly stejného typu (například levá i pravá kulatá), můžeme je kombinovat, toho využijeme například pro vysázení intervalu na ose `\left<0,255\right)` – což znamená, že levá závorka bude ostrá a pravá kulatá, výsledek vypadá takto: $\langle 0, 255)$.

Jestliže chceme vysázet jen jednu ze závorek, pak místo té, která má chybět, napíšeme tečku: `\left|vzorec\right.` (chybí pravá) nebo `\left.vzorec\right)` (chybí levá). Tuto možnost využijeme například tehdy, když na jedné straně výrazu potřebujeme vytvořit svorku, jak vidíme v příkladu výše hned nad nadpisem 4.3.3.

Párování závorek pomocí `\left` a `\right` má především výhodu v tom, že obě závorky jsou stejně velké, ale někdy může působit komplikace. Pak lze využít jiný způsob sázení různě velkých (nejen) závorek – primitiva `\big`, `\Big`, `\bigg`, `\Bigg`, za které píšeme příslušný symbol. Existují také párující primitiva `\bigl`, `\bigr`, `\Bigl`, ..., `\Biggr`. Nevýhodou je, že musíme sami určit, jak vysoký symbol vlastně chceme (což však v některých situacích může být výhodou, pokud při vnořování závorek chceme různé úrovně opticky odlišit a LaTeX samotný to odmítá provést).

Kromě párujících prvků pro pružné závorky máme k dispozici také prvek pro úpravu velikosti centrálního objektu, který využijeme například pro oddělovač v množinovém zápisu:

```
\begin{eqnarray*}
  \lefteqn{
    \mathrm{L}_1=\left\{a^{2n}\bigm| n\geq 0\right\}
  }\\
  & \cup & \left\{b^{3^n}\bigm| n\geq 0\right\}\\
  & \cup & \left\{c^{2^{2n}}\biggm| n\geq 0\right\}
\end{eqnarray*}
```

$$\begin{aligned} \mathrm{L}_1 &= \left\{a^{2n} \mid n \geq 0\right\} \\ &\cup \left\{b^{3^n} \mid n \geq 0\right\} \\ &\cup \left\{c^{2^{2n}} \,\middle|\, n \geq 0\right\} \end{aligned}$$

Výhodou použití modifikátoru `\bigm` a variant pro různé velikosti je, že automaticky přidává kolem vysázeného prvku mezery, které bychom jinak museli přidat ručně. Nevýhodou je, že velikost vysázeného prvku musíme určit sami.

Symboly mnoha „velkých“ unárních operátorů existují ve dvou velikostech podle toho, zda se nacházejí ve vzorci v textu a nebo ve vysazeném prostředí. Horní a dolní index jsou v prvním případě sázeny vedle operátoru, v druhém případě nad a pod operátorem. Pokud toto chování chceme obrátit (například ve vysazeném prostředí mít indexy vedle operátoru místo pod a nad ním), použijeme primitivum `\limits` (vynutí indexy nad a pod operátorem), resp. `\nolimits` (vynutí vysázení indexů vedle operátoru).

Ukázku použití `\limits` najdeme v kódu na straně 47. Jde o vzorec sázený v textu (i když to tak nevypadá), ale chceme, aby horní index byl nad operátorem $\prod$ a dolní index pod ním. Pozor, uvnitř prostředí *array* se dostáváme do řádkového režimu, ve kterém platí nastavení pro vzorce v textu (pokud nepoužijeme `\displaystyle`).

Následuje celkem šest vzorců vždy po dvou na řádku – první dva vysazené, další dva v textu a třetí dvojice v prostředí *array*. Na každém řádku je zobrazeno implicitní chování a pak chování s modifikátorem.

```
$$\sum_{i=0}^n x_i\qquad
  \sum\nolimits_{i=0}^n x_i$$

$\sum_{i=0}^n x_i\qquad
  \sum\limits_{i=0}^n x_i$\centering

$$\begin{array}{c@{\qquad}c}
  \sum_{i=0}^n x_i &
    \sum\limits_{i=0}^n x_i
\end{array}$$
```

$$\sum_{i=0}^{n} x_i \qquad \sum\nolimits_{i=0}^{n} x_i$$

$$\textstyle\sum_{i=0}^{n} x_i \qquad \textstyle\sum\limits_{i=0}^{n} x_i$$

$$\begin{array}{c@{\qquad}c} \sum_{i=0}^{n} x_i & \sum\limits_{i=0}^{n} x_i \end{array}$$

Jak vidíme, druhý a třetí řádek obsahují (alespoň vizuálně) totéž – je to z toho důvodu, že uvnitř prostředí *array* (třetí řádek) přecházíme do řádkového režimu stejně jako u vzorců sázených v textu.

4.3.4 Derivace, integrace

Sazba symbolu derivace je v základu jednoduchá – horní akcent, případně můžeme použít primitivum `\prime` v horním indexu:

```
$$f'(x)=\lim_{h\to 0}\frac{f(x+h)-f(x)}{h}$$
```

$$f'(x)=\lim_{h\to 0}\frac{f(x+h)-f(x)}{h}$$

```
$$f^\prime (x)=\lim_{h\to 0}\frac{f(x+h)-f(x)}{h}$$
```

$$f^\prime (x)=\lim_{h\to 0}\frac{f(x+h)-f(x)}{h}$$

Starší způsob zápisu derivace je s pomocí symbolu diferenciálu d. Protože se nejedná o proměnnou, nesmí být sázen kurzívou. Takže:

```
\def\mathdif{\mathrm{d}}
$$f'(x)=\frac{\mathdif f}{\mahtdif x}$$
```

$$f'(x)=\frac{\mathdif f}{\mahtdif x}$$

V parciální derivaci se místo jednoduchého diferenciálu používá parciální: ∂. Zápis:

```
\begin{eqnarray*}
  \lefteqn{f'_x(x,y) =
    \frac{\partial f}{\partial x}}\\[4pt]
  &=& \lim_{\Delta x\to 0}
    \frac{f(x+\Delta x,y)-f(x,y)}{\Delta x}
\end{eqnarray*}
```

$$\begin{aligned} f'_x(x,y) &= \frac{\partial f}{\partial x} \\ &= \lim_{\Delta x\to 0}\frac{f(x+\Delta x,y)-f(x,y)}{\Delta x} \end{aligned}$$

Pro integrály (všechny jejich variace) existují dvě velikosti (textové a vysazené vzorce). U určitého integrálu si hlídáme umístění horní a dolní meze. Pozor na zápis diferenciálu, nesmí zůstat kurzívou.

U integrálu obvykle použijeme alespoň jednu zápornou mezeru, aby integrovaný výraz nebyl příliš daleko od symbolu integrálu. Naopak před diferenciál přidáváme úzkou mezeru. Další mezery umístíme tak, aby odlišné části vzorce nesplývaly.

```
\begin{eqnarray*}
  F(x) &=& \int \! f(x) \,\mathrm{d}x\\
  \int \! x^n \,\mathrm{d}x
    &=& \frac{x^{n+1}}{n+1}+c\\
  \int_a^b \!\!f(x)\,\mathrm{d}x
    &=& F(b)-F(a)\\
  \int\limits_{a_1}^{a_2}
    \!\int\limits_{b_1}^{b_2}
    \! f(x,y)\,\mathrm{d}y\,\mathrm{d}x
    &=& \int\limits_{a_1}^{a_2}
    \!\left(\int\limits_{b_1}^{b_2}
    \!f(x,y)\,\mathrm{d}y\right)
    \!\mathrm{d}x\\
  \oint\;f(x)\,\mathrm{d}s
    &\mbox{je}&\mbox{křivkový integrál}
\end{eqnarray*}
```

$$
\begin{aligned}
F(x) &= \int f(x)\,\mathrm{d}x \\
\int x^n\,\mathrm{d}x &= \frac{x^{n+1}}{n+1}+c \\
\int_a^b f(x)\,\mathrm{d}x &= F(b)-F(a) \\
\int\limits_{a_1}^{a_2}\int\limits_{b_1}^{b_2} f(x,y)\,\mathrm{d}y\,\mathrm{d}x &= \int\limits_{a_1}^{a_2}\left(\int\limits_{b_1}^{b_2} f(x,y)\,\mathrm{d}y\right)\mathrm{d}x \\
\oint f(x)\,\mathrm{d}s &\quad \text{je} \quad \text{křivkový integrál}
\end{aligned}
$$

V předchozí ukázce jsme viděli také způsob zápisu křivkového integrálu přes uzavřenou křivku (poslední řádek). Další možnosti (dvojný a trojný integrál, plošný integrál apod.) nabízejí různé balíčky, například ams balíčky, dále *wasysym*, *mathabx*, *pxfonts* a další.

4.4 Matematická sazba v teoretické informatice

Dále se podíváme na konvence související se zápisem prvků teoretické informatiky – jazyků, gramatik a automatů.

4.4.1 Množiny a regulární výrazy, jazyky, logika

Jazyk coby množinu slov obvykle zapisujeme buď regulárním výrazem (u regulárních jazyků) nebo množinově. V případě množinového zápisu jde buď o výčet slov a nebo o souhrn kritérií, která jsou splněna právě slovy z daného jazyka.

U množinového zápisu s kritérii (podmínkami) je obvykle nejdřív stanovena forma slov jazyka a pak za oddělovačem následují jednotlivá kritéria. Jako oddělovač se používá buď výše ukázané primitivum `\bigm|` a jeho modifikace, a nebo dvojtečka či středník. Dvojtečka a středník mají výhodu v tom, že nemusíme řešit výšku prvku, ale na druhou stranu kolem primitiva `\bigm|` se automaticky vysázejí mezery, protože jde o operátor. Následuje sekvence příkladů a způsob jejich zápisu v LaTeXu.

- $\mathrm{L}_1 = (ab)^* + (aa+bb)^* = \{\varepsilon, ab, aa, bb, (ab)^2, a^4, a^2b^2, b^2a^2, b^4, \ldots\}$

```
\mathrm{L}_1=(ab)^*+(aa+bb)^*=\left\{\varepsilon,ab,aa,bb,
```

```
(ab)^2,a^4,a^2b^2,b^2a^2,b^4,\dots\right\}
```

- $\mathrm{L}_2 = \left\{w \in \{a,b\}^* \mid w = w^{\mathrm{R}}\right\} = \{\varepsilon, a, b, a^2, b^2, a^4, b^4, abba, baab, \ldots\}$

```
\mathrm{L}_2=\left\{w\in\{a,b\}^*\bigm|w=w^{\mathrm{R}}\right\}
=\left\{\varepsilon,a,b,a^2,b^2,a^4,b^4,abba,baab,\dots\right\}
```

- $\mathrm{L}_2 = \left\{w \in \{a,b\}^* \ ; \ w = w^{\mathrm{R}}\right\}$

```
L_2=\left\{w\in\{a,b\}^*\ ;\ w=w^{\mathrm{R}}\right\}
```

- $\mathrm{L}_3 = \{a^i b^j \ ; \ i \geq 0, \ j \geq 1\}$

```
\mathrm{L}_3=\left\{a^ib^j\ ;\ i\geq 0,\ j\geq 1\right\}
```

- $\mathrm{L}_4 = \{a^i b^j \ ; \ 0 \leq i < j\}$

```
\mathrm{L}_4=\left\{a^ib^j\ ;\ 0\leq i<j\right\}
```

- $\mathrm{L}_5 = \{w \in \{a,b\}^* \ ; \ |w|_a = |w|_b + 1\}$

```
\mathrm{L}_5=\left\{w\in\{a,b\}^*\ ;\ |w|_a=|w|_b+1\right\}
```

- $\mathrm{L}_6 = \{w \in \{a,b\}^* \ ; \ |w| = 2n, \ n \in \mathbb{N}\} \cup \{\varepsilon\} = ((a+b)^2)^*$

```
\mathrm{L}_6=\left\{w\in\{a,b\}^*\ ;\ |w|=2n,\ n\in\mathbb{N}\right\}
\cup\{\varepsilon\}=\left((a+b)^2\right)^*
```

- $\mathrm{L}_7 = \{w \in \{0,1,2\}^* \ ; \ |w|_0 = |w|_1 < |w|_2\}$

```
\mathrm{L}_7=\left\{w\in\{0,1,2\}^*\ ;\ |w|_0=|w|_1<|w|_2\right\}
```

- $\mathrm{L}_8 = a^*, \ \mathrm{L}_9 = a^*b^* \quad \Rightarrow \quad \mathrm{L}_8 \subset L_9$

```
\mathrm{L}_8=a^*,\ \mathrm{L}_9=a^*b^*
\quad\Rightarrow\quad \mathrm{L}_8\subset L_9
```

- $\mathrm{L}_{10} = \left\{wcw^{\mathrm{R}} \ ; \ w \in \{a,b\}^*\right\}$

```
L_{10}=\left\{wcw^\mathrm{R}\ ;\ w\in\{a,b\}^*\right\}
```

- $\mathrm{L}_{11} = \left\{a^{n^2} \ ; \ n \geq 5\right\}$

```
L_{11}=\left\{a^{n^2}\ ;\ n\geq 5\right\}
```

- $\mathrm{L}_{12} = a^*, \ \mathrm{L}_{13} = (a+b)^* - \mathrm{L}_{12} = \{w \in \{a,b\}^* \ ; \ |w|_b \geq 1\}$

```
\mathrm{L}_{12}=a^*,\ \mathrm{L}_{13}=(a+b)^*-\mathrm{L}_{12}
=\left\{w\in\{a,b\}^*\ ;\ |w|_b\geq 1\right\}
```

- $\mathrm{L}_{14} = \{a^p \ ; \ p \text{ je prvočíslo}\}$

```
\mathrm{L}_{14}=\left\{a^p\ ;\ p\ \mbox{je prvočíslo}\right\}
```

- $\mathrm{L}_x \subseteq \mathrm{L}_y \quad \wedge \quad \mathrm{L}_x \supseteq \mathrm{L}_y \quad \Rightarrow \quad \mathrm{L}_x = \mathrm{L}_y$

```
\mathrm{L}_x\subseteq \mathrm{L}_y\quad\wedge\quad \mathrm{L}_x
\supseteq \mathrm{L}_y\quad\Rightarrow\quad \mathrm{L}_x=\mathrm{L}_y
```

- $\mathrm{L} \cup \emptyset = \mathrm{L}, \ \mathrm{L} \cap \emptyset = \emptyset, \ \mathrm{L} \cdot \emptyset = \emptyset$

```
\mathrm{L}\cup\emptyset=\mathrm{L},\ \mathrm{L}\cap\emptyset=\emptyset,
\ \mathrm{L}\cdot\emptyset=\emptyset
```

Co se obecných označení týče, většinou v důkazech používáme

- latinská malá písmena ze začátku abecedy jako terminály (například $\Sigma = \{a, b, c\}$, $w \in \{a, b\}^*$),
- latinská malá písmena z konce abecedy pro označení terminálních řetězců ($w \in \Sigma$, $w = u \cdot v$ nebo v Pumping lemma $w = x \cdot y \cdot z \cdot u \cdot v$),
- latinská velká písmena většinou bývají vyhrazena pro neterminály (A, B, C),
- řecká malá písmena typicky zastupují řetězce, ve kterých mohou být terminály i neterminály ($A \to \alpha \mid \beta$, případně $V_i = V_{i-1} \cap \{X \in (N \cup T) \ ; \ (A \to \alpha X \beta) \in P, \ A \in V_{i-1}\}$).

4.4.2 Gramatiky

Gramatiku, jejíž pravidla chceme mít očíslovaná, zapisujeme takto (pokud nechceme číslovat pravidla, poslední sloupec samozřejmě vynecháme):

```
\newcommand{\krouzek}[1]{\textcircled{\small 1}}%
$G=(\{S,A,B\},\ \{a,b,c\},\ P,\ S)$\\
$\begin{array}{@{}r@{\ \to\ }l@{\ \ \ }r}
  S&AB\bigm|\varepsilon
    &\mbox{\krouzek{1},\krouzek{2}}\\
  A&aAb\bigm|bAa\bigm|ab\bigm|ba
    &\mbox{\krouzek{3},\krouzek{4},
    \krouzek{5},\krouzek{6}}\\
  B&cB\bigm|\varepsilon
    &\mbox{\krouzek{7},\krouzek{8}}
\end{array}$
```

$$
\begin{array}{@{}r@{\ \to\ }l@{\ \ \ }r}
S & AB \mid \varepsilon & \textcircled{1},\textcircled{2} \\
A & aAb \mid bAa \mid ab \mid ba & \textcircled{3},\textcircled{4},\textcircled{5},\textcircled{6} \\
B & cB \mid \varepsilon & \textcircled{7},\textcircled{8}
\end{array}
$$

$G = (\{S, A, B\},\ \{a, b, c\},\ P,\ S)$

Pokud chceme napsat jen jediné pravidlo, pak není nutné využívat prostředí *array*, stačí napsat `A\ \to\ aAb` (vysází se $A \ \to \ aAb$). Místo primitiva `\to` mnozí píšou `\rightarrow` – to je v pořádku, jen je třeba klepnout na o něco víc kláves.

Všimněte si především způsobu, jak jsou oddělena jednotlivá pravidla se stejnou levou stranou (tj. na tomtéž řádku) – svislice je generována primitivem `\bigm|`, který vysázenou svislici automaticky obklopí mezerami. Kdybychom chtěli totéž provést ručně, museli bychom napsat `~|~`.

Zatímco v definici gramatiky používáme pro zápis pravidel *jednoduchou šipku*, při zápisu odvození (derivace) slov píšeme *dvojitou šipku* (ve skutečnosti se jedná o relaci):

$S \Rightarrow AB \Rightarrow aAbB \Rightarrow ababB \Rightarrow \ldots \Rightarrow ababccc$

```
S\Rightarrow AB\Rightarrow aAbB\Rightarrow ababB\Rightarrow\dots\Rightarrow ababccc
```

Pokud potřebujeme přidat informaci o tom, že odvození probíhá podle gramatiky G, zapíšeme $\Rightarrow_G$, tedy `\Rightarrow_G`. Hvězdičku přidáváme stejně jako k jiným objektům, takže $\Rightarrow^*$ vygenerujeme kódem `\Rightarrow^*` (podobně $\Rightarrow_G^*$ vytvoříme kódem `\Rightarrow_G^*`).

4.4.3 Automaty

Označení automatu se obvykle píše kaligrafickým písmem. U Turingova stroje je součástí definice určení značení směru pohybu hlavy, což může být buď čísly nebo písmeny (nevýhoda: musíme sázet vzpřímeným písmem) a nebo šipkami. Následuje ukázka sazby pro konečný a zásobníkový automat a tři varianty zápisu Turingova stroje:

$$\begin{array}{rcll}
\mathcal{A}_K & = & (Q,\ \Sigma,\ \delta,\ q_0,\ F); & \delta\colon\ Q\times\Sigma\to Q \\
\mathcal{A}_Z & = & (Q,\ \Sigma,\ \Gamma,\ \delta,\ q_0,\ Z_0,\ F); & \delta\colon\ Q\times\Sigma\times\Gamma\to Q\times\Gamma^* \\
\mathcal{M}_1 & = & (Q,\ \Sigma,\ \Gamma,\ \delta,\ q_0,\ F,\ \{-1,0,1\}); & \delta\colon\ Q\times\Gamma\to Q\times\Gamma\times\{-1,0,1\} \\
\mathcal{M}_2 & = & (Q,\ \Sigma,\ \Gamma,\ \delta,\ q_0,\ F,\ \{\mathrm{L},\mathrm{S},\mathrm{R}\}); & \delta\colon\ Q\times\Gamma\to Q\times\Gamma\times\{\mathrm{L},\mathrm{S},\mathrm{R}\} \\
\mathcal{M}_3 & = & (Q,\ \Sigma,\ \Gamma,\ \delta,\ q_0,\ F,\ \{\leftarrow,\downarrow,\to\}); & \delta\colon\ Q\times\Gamma\to Q\times\Gamma\times\{\leftarrow,\downarrow,\to\}
\end{array}$$

Předchozí bylo vygenerováno tímto kódem:

```
\def\mrmL{\mathrm{L}}
\def\mrmR{\mathrm{R}}
\def\mrmS{\mathrm{S}}
$$\begin{array}{rclrl}
  \mathcal{A}_K &=&(Q,\ \Sigma,\ \delta,\ q_0,\ F);
    & \delta\colon & Q\times\Sigma\to Q\\
  \mathcal{A}_Z &=&(Q,\ \Sigma,\ \Gamma,\ \delta,\ q_0,\ Z_0,\ F);
    & \delta\colon & Q\times\Sigma\times\Gamma\to Q\times\Gamma^*\\
  \mathcal{M}_1 &=&(Q,\ \Sigma,\ \Gamma,\ \delta,\ q_0,\ F,\ \{-1,0,1\});
    & \delta\colon & Q\times\Gamma\to Q\times\Gamma\times\{-1,0,1\}\\
  \mathcal{M}_2 &=&(Q,\ \Sigma,\ \Gamma,\ \delta,\ q_0,\ F,\ \{\mrmL,\mrmS,\mrmR\});
    & \delta\colon & Q\times\Gamma\to Q\times\Gamma\times\{\mrmL,\mrmS,\mrmR\}\\
  \mathcal{M}_3 &=&(Q,\ \Sigma,\ \Gamma,\ \delta,\ q_0,\ F,\
    \{\leftarrow,\downarrow,\to\});
    & \delta\colon & Q\times\Gamma\to Q\times\Gamma\times\{\leftarrow,\downarrow,\to\}
\end{array}$$
```

V zápisu zpracování slova na vstupu automatem používáme zásadně symbol $\vdash$ generovaný primitivem `\vdash`. V žádném případě zde nepoužíváme šipky, ať už jakékoliv. Postup zpracování slova automatem může být zapsán takto:

$(q_0, abcc, Z_0) \vdash (q_0, bcc, aZ_0) \vdash (q_1, cc, Z_0) \vdash \ldots$

```
(q_0,abcc,Z_0)\vdash(q_0,bcc,aZ_0)\vdash(q_1,cc,Z_0)\vdash\dots
```

Verzi s označením automatu, podle kterého je slovo generováno ($\vdash_{\mathcal{A}}$), vytvoříme kódem `\vdash_\mathcal{A}`, verzi s hvězdičkou $\vdash^*$ kódem `\vdash^*`, jejich kombinace $\vdash^*_{\mathcal{A}}$ je generována kódem `\vdash_\mathcal{A}^*`. O něco estetičtěji vypadá, pokud označení automatu přisuneme blíže k symbolu relace odvození: $\vdash_{\mathcal{A}}$ (`\vdash_{\!\mathcal{A}}`). Všimněte si, že záporná mezera je součástí dolního indexu s označením automatu, celek musí být obklopen složenými závorkami.

U Turingova stroje se v některých zdrojích používají značky zarážek obklopující zpracovávané slovo na pásce – bývá to většinou dvojice \$ (levá zarážka) a # (pravá zarážka), a nebo první uvedený pro obě pozice. Často také najdeme symbol **B** nebo ⊔ (`\sqcup`) označující prázdnou buňku na pásce.

4.4.4 Třídy jazyků

V literatuře najdeme několik různých způsobů zápisu, které však neznamenají zcela totéž. Autor práce by se měl po poradě s vedoucím (školitelem) práce pro jeden rozhodnout a toho se držet v celém textu.

- *Množiny gramatik* patřících do určité třídy se označují příslušnou zkratkou, například FIN = konečné, LIN = lineární, REG = regulární, CF = bezkontextové (context-free), CS = kontextové, často se používá označení číslem v Chomského hierarchii (0 až 3), dále např. 0L představuje bezkontextové Lindenmayerovy systémy.
- Totéž platí pro různé druhy automatů.
- Třídu coby množinu jazyků generovaných určitým typem gramatik značíme
  - $\mathcal{L}$(REG), $\mathcal{L}$(CF), $\mathcal{L}$(0), $\mathcal{L}$(0L) apod. – zapsáno pomocí `\mathcal` (kaligrafickým písmem), zápis představuje množinu jazyků generovaných regulárními, bezkontextovými gramatikami, gramatikami typu 0 a bezkontextovými Lindenmayerovými systémy.
  - $\mathscr{L}$(REG), $\mathscr{L}$(CF), $\mathscr{L}$(0), $\mathscr{L}$(0L) znamená totéž, jen jsme použili `\mathscr` z balíčku *mathrsfs* (tudíž musíme načíst ten balíček).

Pak v definicích, větách a důkazech používáme například

```
\mathscr{L}(\mbox{FIN})\subset\mathscr{L}(\mbox{REG})\subset\mathscr{L}(\mbox{CF})
  \subset\mathscr{L}(\mbox{CS})\subset\mathscr{L}(0)
```

výsledek: $\mathscr{L}(\mathrm{FIN}) \subset \mathscr{L}(\mathrm{REG}) \subset \mathscr{L}(\mathrm{CF}) \subset \mathscr{L}(\mathrm{CS}) \subset \mathscr{L}(0)$

```
\mathscr{L}(\mbox{CF})\not\subseteq \mathscr{L}(0\mathrm{L}),\
  \mathscr{L}(0\mathrm{L})\not\subseteq\mathscr{L}(\mbox{CF})
```

výsledek: $\mathscr{L}(\mathrm{CF}) \not\subseteq \mathscr{L}(0\mathrm{L}),\ \mathscr{L}(0\mathrm{L}) \not\subseteq \mathscr{L}(\mathrm{CF})$

```
$$\begin{array}{rcl}
\lefteqn{L_1=\left\{a^{2^n}\ ;\ n\geq 0\right\}}\\
  &\not\in&\mathscr{L}(\mbox{CF})\\
  &\in&    \mathscr{L}(0\mathrm{L})\\
  &\in&    \mathscr{L}(0)
\end{array}$$
```

$$\begin{array}{rcl} \lefteqn{L_1=\left\{a^{2^n}\ ;\ n\geq 0\right\}} \\ &\not\in& \mathscr{L}(\mathrm{CF}) \\ &\in& \mathscr{L}(0\mathrm{L}) \\ &\in& \mathscr{L}(0) \end{array}$$

4.5 Definice, věty, důkazy

V práci lze využívat běžná prostředí pro definice, věty a další podobné prvky. V souboru `config.tex` jsou definována prostředí *definice*, *veta*, *lemma* a *dusledek*, a to tak, že jejich číslování je závislé na číslu kapitoly.

Definice 4.1 (Definovaný pojem) *Toto je text definice, ve kterém řádně definujeme pojem, který jsme uvedli v nepovinném parametru. Pokud definici odněkud přejímáme, měl by být v tomto nepovinném parametru uveden i zdroj.*

Výše uvedená definice byla vygenerována kódem

```
\begin{definice}[Definovaný pojem]
Toto je text definice, ... uveden i~zdroj.
\end{definice}
```

Na všechny (číslované) definice, věty apod. se lze v textu odkazovat. Předpokladem je, že v těle prostředí (ideálně hned na začátku) použijeme makro `\label`, kterým jednoznačně označíme dané prostředí konkrétním štítkem (labelem). Ukázku vidíme na následující větě:

Věta 4.1 *Pro věty a lemmata (tj. „menší věty“) jsou určena prostředí* veta *a* lemma. *Stejně jako definice, i zde je text sázen kurzívou. Všimněte si, že názvy těchto prostředí (s makrem* `\emph`*) jsou sázeny inverzně (tj. ne kurzívou).*

Výše uvedená věta byla vygenerována kódem

```
\begin{veta}\label{th:ukazkalemmatu}
Pro věty a~lemmata (tj. \uv{menší věty}) jsou určena prostředí
\emph{veta} a~... (s~makrem {\small\verb,\emph,) jsou sázeny inverzně...
\end{veta}
```

Na tuto větu můžeme odkázat následovně: „Ve větě 4.1...“, což představuje kód

`Ve větě \ref{th:ukazkalemmatu}`. Můžeme přidat i odkaz na stranu založený na stejném štítku (kódem `na straně \pageref{th:ukazkalemmatu}`). Aby odkaz fungoval, musí být makro `\label` alespoň jednou přeloženo.

Lemma 4.2 (Pumping Lemma [3]) *Toto je ukázka lemmatu s názvem a odkazem do seznamu použité literatury. Samotné Pumping lemma samozřejmě vypadá úplně jinak, zde pouze demonstrujeme, jak má vypadat citované lemma. Všimněte si, že lemmata a věty používají číslování v téže řadě.*

Výše uvedené lemma bylo vygenerováno kódem

```
\begin{lemma}[Pumping Lemma \cite{lengths}]
Toto je ukázka lemmatu ... řadě.
\end{lemma}
```

Samozřejmě zápis citace odpovídá tomu, že v seznamu použité literatury je položka

```
\bibitem{lengths}
Lengths and when to use them [online].
\emph{StackXchange}  [cit. 2013-12-04]  Dostupné na:\\
\iadresa{\http://tex.stackexchange.com/questions/41476/lengths-and-when-to-use-them}
```

Důkaz: Důkaz uvedené věty, lemmatu, důsledku, tvrzení apod. se narozdíl od jmenovaných nesází kurzívou. Důkaz ukončujeme symbolem „boxu“, což je obvykle prázdný čtvereček. □

Výše uvedený text důkazu byl vygenerován kódem

```
\begin{dukaz}
Důkaz uvedené věty, ... čtvereček.
\end{dukaz}
```

4.6 Zápis příkladů

V souboru `config.tex` jsou definovaná další prostředí, která se můžou v určitých typech prací hodit.

Příklad 4.1 Zde napíšeme text příkladu. Každý příklad je číslován (v závislosti na čísle kapitoly) a je možné na něj odkazovat stejným způsobem jako na definici nebo větu.

Na příkladu 4.1 vidíme, jak použít prostředí *priklad*. Odkaz na příklad byl vygenerován kódem `\ref{pri:ukazkaprikladu}`, samotný příklad je generován kódem

```
\begin{priklad}\label{pri:ukazkaprikladu}%
Zde napíšeme text příkladu. Každý ...
\end{priklad}
```

Všimněte si, že za makrem `\label` je komentářový symbol. Bez toho by byl první řádek vnitřního textu mírně odsazen, což by nevypadalo hezky.

Nevýhodou klasického vzhledu příkladu je to, že na první pohled nepoznáme, kde příklad končí, zvláště pokud je uvnitř prostředí převážně text. Pokud trváme na zřetelnějším vyznačení začátku a konce příkladu, můžeme použít následující prostředí:

Příklad 4.2

Toto prostředí je číslováno, navíc máme zřetelně vyznačen začátek a konec prostředí (pomocí horizontálních čar). Nevýhodou prostředí je to, že je nemůžeme použít v zúžené sazbě (například tehdy, když vložíme obtékaný obrázek), někdy také dělá problémy při použití v kombinaci se seznamy (číslovanými, odrážkovými apod.).

Na prostředí se odkazujeme stejně jako v předchozím případě: „Příklad 4.2 ukazuje…“. Celé prostředí bylo vygenerováno kódem

```
\begin{prikladcary}\label{pri:prikladscarami}%
Toto prostředí je číslováno, navíc ...
\end{prikladcary}
```

4.7 Zápis úseků programového kódu

V souboru `config.tex` najdeme prostředí pro programový kód *prog* využívající balíček *alltt* (proto tento balíček musíme přidat do preambule hlavního dokumentu, aby prostředí fungovalo).

Výhodou je, že v prostředí lze snadno nastavit jiné řádkování. Zde například máme použito řádkování 1 (možná by bylo úpravnější řádkování 1.2, ale určitě ne 1.5 jako v běžném odstavci), které je pro neproporcionální písmo lepší. Nevýhodou je nutnost dávat opačné lomítko před všechno, co by LaTeX mohl chtít interpretovat (například před lomené závorky), pro zápis zpětného lomítka je nutné použít příkaz

`\zpetnelomitko` (je definován tamtéž). Celé prostředí doporučuji obklopit prázdnými řádky.

Zde vidíme úsek kódu v Javě:

```
// třída pro zpracování příkazu CACLS
public class TCacls {
    TData data;
    public String out;  ...
```

Bylo vygenerováno následujícím kódem:

```
\begin{prog}
// třída pro zpracování příkazu CACLS
public class TCacls \{
    TData data;...
\end{prog}
```

Všimněte si, že před složenou závorku je vepsáno zpětné lomítko. Když srovnáte předchozí zápis a jeho kód, zjistíte, že v prostředí *prog* bylo použito řádkování 1, nikoliv řádkování 1,5.

Obecně můžeme pro zápis kódu využít buď to, co bylo výše ukázáno, nebo jednoduše prostředí *verbatim*, případně prostředí *alltt* ze stejnojmenného balíčku (na něm je postaveno výše uvedené prostředí *prog*), a nebo lze využít další speciální balíčky. Například velmi úpravné zápisy algoritmů lze vytvářet pomocí prostředí *algorithm* z balíčku *algorithm2e*. Tento balíček může za určitých okolností způsobit kolizi s balíčkem *hyperref*, důsledkem je chybné zobrazování menu v prohlížečích .PDF souborů.

```
\ begin{algorithm}[H]
příchod ke křižovatce\;
\Repeat{nejede auto}
{
  pohlédni vlevo\;
  \While{jede auto}{čekej\;}
  pohlédni vpravo\;
  \While{jede auto}{čekej\;}
  pohlédni vlevo\;
}
přejdi\;
\ caption{Přechod křižovatky}
\ label{alg:krizovatka}
\ end{algorithm}
```

Algoritmus 1: Přechod křižovatky

příchod ke křižovatce;
repeat
  pohlédni vlevo;
  while *jede auto* **do**
    čekej;
  pohlédni vpravo;
  while *jede auto* **do**
    čekej;
  pohlédni vlevo;
until *nejede auto*;
přejdi;

Je zřejmé, že toto řešení je ideální především pro zápis kratšího pseudokódu, který se vejde na jednu stranu, výhodou je automatické zvýrazňování klíčových slov.

Algoritmy v prostředí *algorithm* jsou vlastně plovoucí prostředí stejně jako obrázky a tabulky, a jak vidíme na kódu, můžeme se na ně také odkazovat, například

```
V algoritmu \ref{alg:krizovatka} je uvedeno
```

Také se lze odkazovat na jednotlivé řádky, pokud jsou číslovány, navíc můžeme ne-

chat automaticky vygenerovat seznam algoritmů v podobném stylu jako seznamy obrázků a tabulek. Tuto funkčnost stejně jako další možnosti najdeme v dokumentaci k balíčku.

```
\LinesNumbered
\begin{algorithm}[H]
vezmi knihu\;
otevři knihu na první straně\;
\While{není konec knihy}{
  čti kapitolu\;
  \eIf{rozumíš kapitole}{
    přesuň se na další kapitolu\;
    \If{víš kdo je vrah}{
      přestaň číst\;}
      přesuň se na konec knihy\;}
  }{
    zpět na začátek kapitoly\;}
  }
\caption{Hledání vraha}
\label{alg:hledanivraha}
\end{algorithm}
```

Algoritmus 1: Hledání vraha

```
1  vezmi knihu;
2  otevři knihu na první straně;
3  while není konec knihy do
4  |  čti kapitolu;
5  |  if rozumíš kapitole then
6  |  |  přesuň se na další kapitolu;
7  |  |  if víš kdo je vrah then
8  |  |  |  přestaň číst;
9  |  |  |  přesuň se na konec
   |  |  |  knihy;
10 |  else
11 |  |  zpět na začátek kapitoly;
```

Vlastnosti zobrazení lze řídit nepovinnými parametry při načítání balíčku,

například můžeme uvést: `\usepackage[ruled,vlined,czech]{algorithm2e}` (použito pro výše uvedené algoritmy), čímž zapneme čárové zvýrazňování vnořování a ukončení bloků a zápis v češtině.

4.8 Stromy, diagramy, grafy a jiná užitná grafika

Grafické objekty lze vytvářet buď externě (grafický editor, například MSPaint, Visio, Gimp, Photoshop, Paint.NET, Corel, Inkscape, některý CAD systém, atd. – omlouvám se, pokud zde není zrovna váš oblíbený) a pak vložit pomocí `\includegraphics`, a nebo přímo s využitím balíčků LaTeXu.

Je možné oba postupy kombinovat (sice s využitím LaTeXu, ale vytvořit soubor ve formátu .EPS nebo .PDF a pak vložit pomocí `\includegraphics`), což využijeme zejména tehdy, když zvolený grafický balíček vyžaduje jiný typ překladu než jaký používáme pro celý dokument, a nebo když se vyskytují další problémy při překladu.

Existují grafické programy generující přímo TeXový kód – *LaTeXPiX*, *TeXcad32*, *TeXCAD*, atd.

Konverze souborů, které chceme vkládat, lze provést v různých programech (Gimp, ImageMagick, XnView, atd.), liší se však kvalita této konverze. Soubor ve

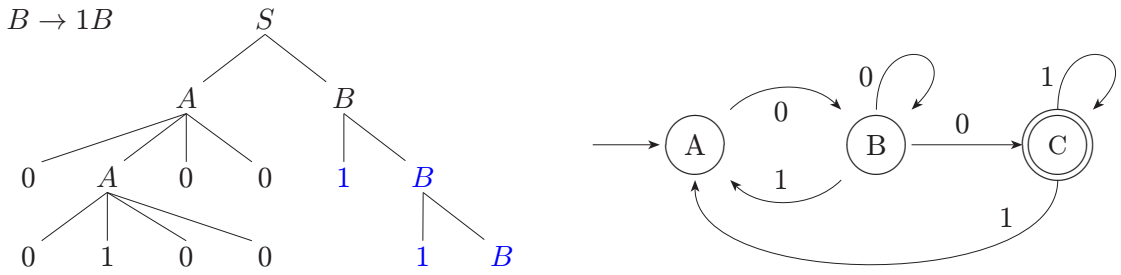
formátu .PS můžeme přeložit do formátu .EPS přímo v prohlížeči *GSView* (*File – PS to EPS*, necháme zatrženou volbu „Automaticaly calculate Bouding Box“). Vkládání obrázků a jiných objektů je popsáno v tomto dokumentu od strany 36.

4.8.1 Stromy, diagramy, grafické struktury

Pokud chceme zobrazit derivační strom, diagram konečného automatu či jakoukoliv jinou 2D strukturu, potřebujeme aparát na kreslení uzlů a spojnic mezi nimi včetně jejich ohodnocování. K tomu účelu existuje více různých balíčků, zde si ukážeme použití balíčku *tree-dvips*.

Už z názvu vidíme, že zdroj má být překládán do formátu .PS. Pokud celý dokument překládáme programem pdfTEX, vytvoříme obrázky v samostatných souborech, přeložíme do .PS, konvertujeme do .EPS (viz výše) a vložíme pomocí `\includegraphics`.

V příloze B jsou dva příklady struktur vytvořených pomocí *tree-dvips* – derivačního stromu a diagramu konečného automatu. Obojí vidíme na obrázku 4.



Obrázek 4: Ukázky nákresů vytvořených pomocí balíčku *tree-dvips*

To je jen krátká ukázka schopností tohoto balíčku. S jeho pomocí můžeme vytvářet také spojnice se zalomením, lze určovat typy a rozměry šipek či jiných zakončení apod. V závěrečných pracech se vyskytují často i další typy struktur: use-case diagram, Ishikawův diagram (příčin a následků) apod., i pro tyto účely se dá využít balíček *tree-dvips*. Informace najdeme v dokumentaci balíčku.

Alternativy: balíčky *xyling*, *trees*, *qtree*, *ps-trees* a další.

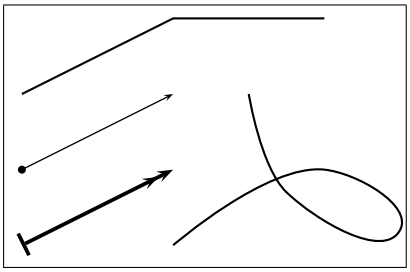
4.8.2 Grafy a jiné grafické prvky

V předchozím textu jsme se zabývali jednoduchými spojnicovými grafy, ale grafy mohou být i složitější. Pro kreslení dalších typů grafů máme k dispozici další balíčky

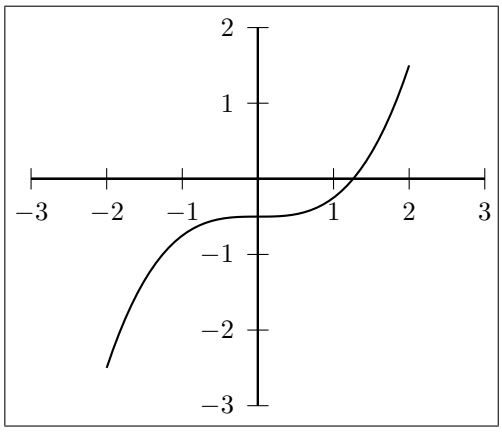
(pokud se nerozhodneme pro vytvoření v jiném programu).

Balíček *pstricks* je velmi komplexní nástroj pro vytváření prakticky jakékoliv vektorové grafiky. Dále si ukážeme použití některých příkazů balíčku *pstricks*. Jak název napovídá, opět budeme překládat přes formát .PS. Samotný balíček nabízí spoustu příkazů souvisejících s grafikou, my si zde ukážeme jen to nejzákladnější – čáry s různými zakončeními a křivku. Máme k dispozici prostředí `pspicture` podobné prostředí `picture`, předdefinované jednotky jsou však centimetry, ne body.

```
\documentclass{article}
\usepackage{pstricks}
\pagestyle{empty}
\begin{document}
\begin{pspicture}(6,3)
  \psline(0,2)(2,3)(4,3)
  \psline[arrows=*->,linewidth=.5pt](0,1)(2,2)
  \psline[arrows=|->>,linewidth=.5mm](0,0)(2,1)
  \pscurve(2,0)(4,1)(5,.2)(3.5,.7)(3,2)
\end{pspicture}
\end{document}
```



```
\documentclass{article}
\usepackage{pstricks}
\usepackage{pst-plot}
\pagestyle{empty}
\begin{document}
\begin{pspicture}(4,2)
  \psaxes(0,0)(-3,-3)(3,2)
  \psplot{-2}{2}{x 3 exp 4 div 0.5 sub}
\end{pspicture}
\end{document}
```



Další možnosti nám poskytne kombinace balíčku *pstricks* a některého dalšího, například *pst-plot* (tj. příkazem `\usepackage` načteme nejdřív *pstricks* a pak ještě *pst-plot*), v našem případě máme k dispozici kromě jiného příkazy k vykreslování funkcí. Graf jedné funkce je na předchozím obrázku. První příkaz vykreslí soustavu souřadnic, druhý příkaz funkci $x^3/4 - 0.5$ na intervalu $x \in \langle -2, 2 \rangle$ (to jsou první dva parametry).

Zápis funkce je v postfixu, kdy se nejdřív píšou oba operandy a pak název funkce nebo operátor, například $4 + 5$ se zapíše jako $4\ 5\ +$ nebo výraz $2 * x + 4$ jako $2\ x\ *\ 4\ +$. Postfixový zápis vytvoříme jednoduše: nejdřív si „uzávorkujeme“ celý výraz, například $a + 2 * x + 4 - z$ uzávorkujeme na $(((a + (2 * x)) + 4) - z)$, postupujeme podle toho, které operátory mají přednost, v případě několika se stej-

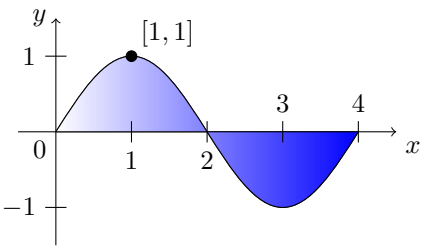
nou předností (třeba +,-) postupujeme zleva. Pak jdeme od nejvnitřnějších závorek – nejdřív napíšeme 2 x $*$, pak řešíme $+$ s výsledkem a 2 x $*$ $+$ (operátor přidáváme na konec, operandy podle pořadí v původním výrazu), atd., v našem případě bude výsledek „a 2 x $*$ $+$ 4 $+$ z $-$“.

Tento balíček nelze používat zároveň s balíčkem *color*, což ale nevadí, protože má vlastní (a dokonalejší) správu barev.

Ukázky toho, co lze vytvořit s *pstricks*, vidíme i přímo v manuálech na

- http://ftp.cvut.cz/tex-archive/graphics/pstricks/base/doc/pstricks-doc.pdf
- http://www.tex.uniyar.ac.ru/doc/pst_ug.pdf
- http://sarovar.org/projects/pstricks/

Další balíček, kterému bychom měli věnovat pozornost, je *tikz*. Ve skutečnosti se jedná o frontend balíčku *pgf* (zkratka z „portable graphics format“). Jedná se opět o rozsáhlý systém pro vytváření kvalitní vektorové grafiky. Oproti *pstricks* má *tikz* kompaktnější syntaxi a zdroj nemusíme překládat přes Postscript (ovšem je třeba použít příslušný parametr – *pdftex*). Na tomto místě opět nemá smysl popisovat všechny možnosti balíčku, uvedeme si jen krátkou ukázku.




```
\documentclass[dvips]{article}
\usepackage{xcolor}
\usepackage{tikz}
\pagestyle{empty}

\begin{document}
\begin{tikzpicture}
  % souradnicovy system
  \draw[->] (-0.5,0) - (4.5,0) node[below right] {$x$};
  \draw[->] (0,-1.5) - (0,1.5) node[left] {$y$};
  \node[below left] at (0,0) {$0$};
  % funkce:
  \draw (0,0) sin (1,1) cos (2,0) sin (3,-1) cos (4,0);
  \shadedraw[left color=white,right color=blue]
    (0,0) sin (1,1) cos (2,0) sin (3,-1) cos (4,0) - (0,0) -cycle;
  \filldraw (1,1) circle(2pt);
  \node[above right] at (1,1) {$[1,1]$};
  % na ose x
  \draw (1,4pt)-(1,-4pt) node[below] {$1$};
  \draw (2,4pt)-(2,-4pt) node[below] {$2$};
  \draw (3,-4pt)-(3,4pt) node[above] {$3$};
  \draw (4,-4pt)-(4,4pt) node[above] {$4$};
  % na ose y
  \draw (4pt,1)-(-4pt,1) node[left] {$1$};
  \draw (4pt,-1)-(-4pt,-1) node[left] {$-1$};
\end{tikzpicture}
\end{document}
```

Další ukázky kódu jsou v příloze B.3 od strany iv. Navíc je možné vykreslovat i funkce zadané polynomem, s využitím logaritmu, goniometrických funkcí apod. – cokoliv, čemu rozumí GNUPlot (je nutné použít parametr *plot*, viz dokumentace). Základní nabídku *pgf* a *tikz* můžeme rozšířit pomocí speciálních knihoven, existuje například knihovna pro kreslení L-Systémů, konečných automatů, knihovna pro Petriho sítě, myšlenkové mapy, pro želví grafiku, zvětšování úseků obrázku a další.

Balíček *pgf* umožňuje i další praktickou věc – pokud tentýž obrázek chceme zobrazit na více místech v dokumentu, lze jej fyzicky do souboru vložit jen jednou a na příslušných místech (vícekrát) je sice zobrazen, ale reprezentován jen jako odkaz na původní místo vložení. Výsledný soubor je pak menší.

Další informace o těchto balíčcích včetně příkladů najdeme na adresách

- http://ftp.math.purdue.edu/mirrors/ctan.org/graphics/pgf/base/doc/generic/pgf/pgfmanual.pdf
- http://www.texample.net/tikz/examples/author/pgf-manual/

Závěr

Úvod, Závěr a Seznam použité literatury jsou nečíslované kapitoly řazené do Obsahu.

Do Závěru píšeme souhrn poznatků zjištěných v práci, hodnotíme výsledek práce (někde by tu měla být i větička „Cíl práce byl splněn“, nějak vhodně rozvedená). Také tu můžeme psát o případných problémech, se kterými jsme se při psaní práce setkali, možnostech využití, námětech na pokračování do budoucna (tj. co by se dalo zlepšit, přidat, rozšířit, atd.).

Seznam použité literatury

[1] *Generátor citací* [online]. [cit. 2012-06-06]. Dostupné z: https://www.citacepro.com/

[2] KNUTH, Donald. *TEXbook – the Source of the TEXbook* [online]. *CTAN*, 1984 [cit. 2012-06-06]. Dostupné z: http://www.ctan.org/pkg/texbook

[3] Lengths and when to use them [online]. *StackXchange* [cit. 2013-12-04]. Dostupné z: http://tex.stackexchange.com/questions/41476/lengths-and-when-to-use-them

[4] *Metodický pokyn pro úpravy, zveřejňování a ukládání závěrečných prací posluchačů Slezské univerzity v Opavě, Filozoficko-přírodovědecké fakulty v Opavě* [online]. Revize platná od 1. 10. 2010. Opava: FPF SU v Opavě, 2010 [cit. 2013-01-20]. Dostupné z: https://www.slu.cz/file/cul/26d53de1-b1dd-48ab-9cd9-8c4dd3b89b01

[5] *Metodický pokyn rektora K odevzdávání, ukládání a zveřejňování závěrečných prací a habilitačních prací Slezské univerzity v Opavě* [online]. Opava: SU v Opavě, 2017 [cit. 2021-06-01]. Dostupné z: https://www.slu.cz/file/cul/fb6d8286-367b-4ebb-b373-cc0410c5b3c3

[6] OLŠÁK, Petr. *TEXbook naruby*. Brno: Konvoj, 2001. ISBN 80-7302-007-6. Také dostupné z: ftp://math.feld.cvut.cz/olsak/tbn/tbn.pdf

[7] PAKIN, Scott. *The Comprehensive LATEX Symbol List* [online]. 2009 [cit. 2014-01-21]. Dostupné z: http://www.tex.ac.uk/tex-archive/info/symbols/comprehensive/symbols-a4.pdf

[8] *Pokyny pro zadávání kvalifikačních prací (bakalářské, diplomové)* [online]. Opava: SU v Opavě, 2017 [cit. 2021-06-01]. Dostupné z: https://www.slu.cz/fpf/cz/layout/1657

Seznam obrázků

Seznam tabulek

Seznam zkratek

IS	Information System
LAN	Local Area Network
MAC	Media Access Control
TAB	tabulátor
TCP	Transmission Control Protocol

Appendices

Do tohoto seznamu napište přílohy vložené přímo do této práce a také seznam elektronických příloh, které se vkládají přímo do archivu závěrečné práce v informačním systému zároveň se souborem závěrečné práce. Elektronickými přílohami mohou být například soubory zdrojového kódu aplikace či webových stránek, předpřipravený produkt (spustitelný soubor, kontejner apod.), vytvořená metodická příručka, tutoriál... (tento text odstraňte)

- Přílohy v souboru závěrečné práce:
  - Příloha A xxxx
  -
- Elektronické přílohy:
  - Příloha A xxxx
  -

A První příloha

Také přílohy mohou být členěny do podkapitol různých úrovní.

Do přílohy dáváme například:

- rozměrné obrázky, které nemusejí být až tak nutně v textu, moc se na ně neodkazujeme,
- ukázky zdrojového kódu delší než jen pár řádků, pokud je opět z nějakého důvodu nepotřebujeme přímo v textu,
- texty, které se netýkají přímo tématu práce, ale mohou je rozvíjet,
- atd.

B Zdrojové kódy delších příkladů

B.1 Derivační strom

Soubor `derivacnistrom.tex` ve složce `pictures` obsahuje tento kód:

```
\documentclass{article}
\usepackage{czech,tree-dvips,color}

\pagestyle{empty}
```

Číslo strany nesmí být na obrázku.

```
\def\mezera{\rule{1.5em}{0pt}}
```

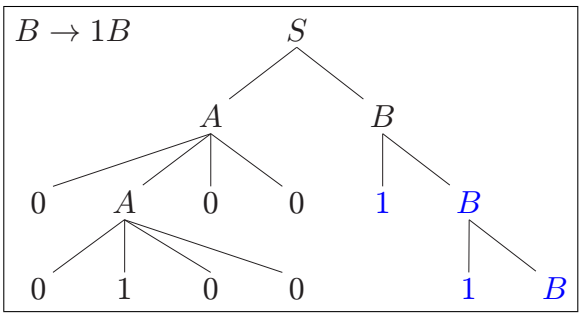
Kvůli zarovnání, šířka sloupce

```
\begin{document}
\begin{tabular}{cccccccc}
\mezera&\mezera&\mezera&\mezera&\mezera&\mezera&\mezera\\
\multicolumn{3}{l}{$B\rightarrow 1B$}&\node{s}{$S$}&\\[1.5em]
&&\node{a1}{$A$}&&\node{b1}{$B$}&\\[1.5em]
\node{01}{$0$}&\node{a2}{$A$}&\node{02}{$0$}&\node{03}{$0$}
  &\color{blue}\node{11}{$1$}&\color{blue}\node{b2}{$B$}&\\[1.5em]
\node{04}{$0$}&\node{12}{$1$}&\node{05}{$0$}&\node{06}{$0$}
  &&\color{blue}\node{13}{$1$}&\color{blue}\node{b3}{$B$}&
\end{tabular}
```

Protože chceme mít uzly zarovnané, umístíme je do tabulky.

Každý uzel má dva parametry – první je *název*, druhý pak zobrazovaný řetězec (takže `\node{název}{řetězec}`). Název využíváme při kreslení spojnic pomocí `\nodeconnect`.

```
\nodeconnect[b]{s}[tr]{a1}
\nodeconnect[b]{s}[tl]{b1}
\nodeconnect[b]{a1}[tr]{01}
\nodeconnect[b]{a1}[tr]{a2}
\nodeconnect[b]{a1}[t]{02}
\nodeconnect[b]{a1}[tl]{03}

\nodeconnect[b]{a2}[tr]{04}
\nodeconnect[b]{a2}[t]{12}
\nodeconnect[b]{a2}[t]{05}
\nodeconnect[b]{a2}[tl]{06}

\nodeconnect[b]{b1}[t]{11}
\nodeconnect[b]{b1}[tl]{b2}
\nodeconnect[b]{b2}[t]{13}
\nodeconnect[b]{b2}[tl]{b3}
\end{document}
```



Formát makra `\nodeconnect`:

```
\nodeconnect[místo u prvního uzlu]{název prvního uzlu}
            [místo u druhého uzlu]{název druhého uzlu}
```

Jednotlivé možnosti umístění začátku a konce spojnice:

- vertikální
  - `t` – top, horní okraj
  - `b` – bottom, spodní okraj
- horizontální
  - `l` – left, levý okraj
  - `r` – right, pravý okraj

Píšeme buď jednu z těchto možností, a nebo kombinujeme v pořadí (vertikální, horizontální), což představuje jednotlivé rohy pomyslného obdélníka kolem uzlu (například `tr` znamená horní pravý roh).

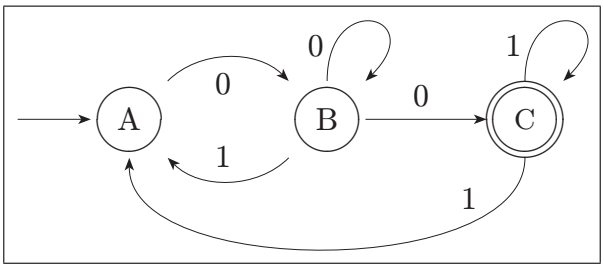
B.2 Diagram konečného automatu

Ukázku kódu diagramu konečného automatu najdeme v souboru `diagram.tex` ve složce `pictures`. Oproti předchozí ukázce zde sice nejsou použity barvy, ale na druhou stranu vidíme vytvoření jednoduchých orientovaných hran, které navíc jsou buď rovné a nebo tvořené křivkou. Hrany jsou ohodnoceny. Výsledkem je tedy ohodnocený orientovaný graf.

```
\documentclass{article}
\usepackage{czech,tree-dvips}
\pagestyle{empty}

\newcommand{\uzel}[2]{\node{#1}{\begin{picture}(20,20)
    \put(10,10){\circle{20}}
    \put(0,0){\makebox(20,20){#2}}
  \end{picture}}}
\newcommand{\uzelf}[2]{\node{#1}{\begin{picture}(20,20)
    \put(10,10){\circle{20}}
    \put(10,10){\circle{22}}
    \put(0,0){\makebox(20,20){\small #2}}
  \end{picture}}}
\newcommand{\uzelempty}{\begin{picture}(20,20)\end{picture}}
\newcommand{\uzelprvni}[1]{\node{#1}{\rule{0pt}{20pt}}}

\newcommand{\ohodnot}[3]{\begin{picture}(0,0)\put(#1,#2){3}\end{picture}}
\def\mezera{\rule{5em}{0pt}}

\begin{document}
\begin{tabular}{cccc}
  &\mezera&\mezera&\mezera\\
  \uzelprvni{prvni}&\uzel{0}{A}\ohodnot{17}{18}{0}\ohodnot{17}{-5}{1}
  &\uzel{1}{B}\ohodnot{17}{14}{0}\ohodnot{-16}{30}{0}
  &\uzelf{2}{C}\ohodnot{-16}{30}{1}\ohodnot{-30}{-18}{1}
\end{tabular}

\anodeconnect[r]{prvni}[l]{0}
\anodecurve[tr]{0}[tl]{1}{1.4em}
\anodecurve[bl]{1}[br]{0}{1.4em}
\anodeconnect[r]{1}[l]{2}
\anodecurve[b]{2}[b]{0}{3.8em}
\anodecurve[t]{2}[tr]{2}{.4in}
\anodecurve[t]{1}[tr]{1}{.4in}
\end{document}
```

Vytvoříme makra pro různé typy uzlů (stavů) – běžný, koncový stav, neviditelný, pomocný první uzel. Každý viditelný uzel je uzavřen v kružnici, konečný ve dvojité.

Ohodnocení hran (první dva parametry jsou souřadnice, třetí parametr zobrazovaný řetězec).



Formát makra `\(a)nodecurve`:

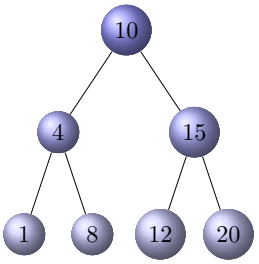
```
\anodecurve[místo u prvního uzlu]{název prvního uzlu}
           [místo u druhého uzlu]{název druhého uzlu}
           {míra zakřivení}
```

Makra generující spojnice, která začínají písmenem „a“, vytvoří u cílového uzlu šipku. Makra bez písmene „a“ na začátku ji nevytvoří. U spojnice ve tvaru křivky musíme oproti jednoduché rovné spojnici zadat i parametr určující míru zakřivení.

B.3 Několik ukázek využití balíčku *tikz*

B.3.1 Jednoduchá struktura – strom

První příklad této části demonstruje způsob vytvoření struktury s uzly (v tomto případě binární). Kód najdeme v souboru `tikzpriklad3.tex`.




```
\documentclass[dvips]{article}
\usepackage{xcolor}
\usepackage{tikz}
\pagestyle{empty}

\begin{document}
\begin{tikzpicture}
  [every node/.style={circle,ball color=blue!40!white},
level 1/.style={sibling distance=20mm,nodes={ball color=blue!30!white}},
level 2/.style={sibling distance=10mm,nodes={ball color=blue!20!white}}]
\node {10}
  child { node {4}
    child {node {1}}
    child {node {8}}
  }
  child { node {15}
    child {node {12}}
    child {node {20}}
  };
\end{tikzpicture}
\end{document}
```

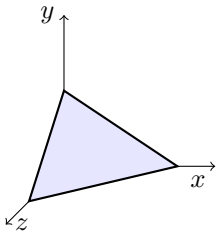
Nastavíme vlastnosti uzlů na úrovních 1 a 2 (všechny uzly kulaté, jednotlivé úrovně se liší horizontálním odsazením a barvou). Úroveň 0 je kořen stromu.

Pokud jsou uvnitř `\node` prvky `child`, znamená to, že uzel má potomky.

Vlastnosti uzlů na jednotlivých úrovních lze nastavovat i přímo ve struktuře, nemusí jít o vlastnosti definované pro celé prostředí *tikzpicture*.

B.3.2 3D souřadnice

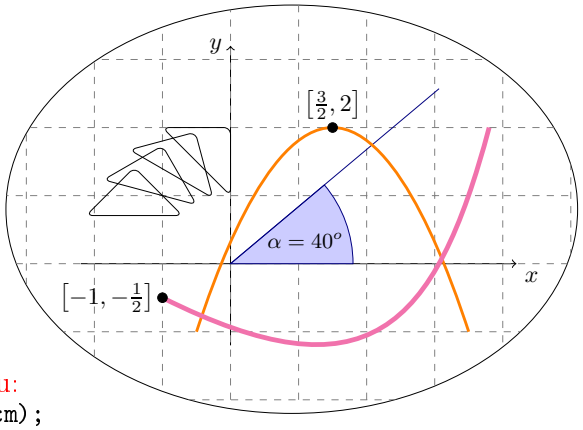
Na dalším obrázku vidíme práci v 3D prostoru. Používáme 3D souřadnice a můžeme je kombinovat se souřadnicemi 2D (pro $z = 0$). Kód je v souboru `tikzpriklad4.tex`.




```
\documentclass[dvips]{article}
\usepackage{xcolor}
\usepackage{tikz}
\pagestyle{empty}

\begin{document}
\begin{tikzpicture}[->]
\draw (0,0) - (2,0) node[below left] {$x$};
\draw (0,0) - (0,2,0) node[left] {$y$};
\draw (0,0) - (0,0,2) node[right] {$z$};
\draw[thick, fill=blue!10] (1.5,0,0) - (0,1,0) - (0,0,1.2) - cycle;
\end{tikzpicture}
\end{document}
```

B.3.3 Úhly, křivky, otáčení

Třetí příklad nám ukazuje způsob vytvoření úhlu, otáčení úsečky a celého objektu, vykreslení souřadné mřížky, paraboly a Bézierovy křivky. Celý obrázek je ořezán do tvaru elipsy (hned první příkaz v prostředí). Kód je v souboru `tikzpriklad2.tex`.




```
\documentclass[dvips]{article}
\usepackage{xcolor}
\usepackage{tikz}
\pagestyle{empty}

\begin{document}
\begin{tikzpicture}
\clip[draw] (0.9,0.8) ellipse (4.2cm and 3cm);

\draw[->] (-2.2,0) - (4.2,0) coordinate (x axis) node[below right] {$x$};
\draw[->] (0,-1.2) - (0,3.2) coordinate (y axis) node[left] {$y$};
\draw[help lines, dashed,step=1] (-4,-2) grid (5,4);

\filldraw[fill=blue!20,draw=blue!50!black] (0,0)-(18mm,0) arc (0:40:18mm) -cycle;
\node[above] at (1.1,0.1) {\small $\alpha=40^o$};
\draw[color=blue!50!black, rotate=40] (0,0)-(4,0);

\draw[very thick,color=orange] (-0.5,-1) parabola bend (1.5,2) (3.5,-1);
\filldraw (1.5,2) circle (2pt) node[above] {$\left[\frac{3}{2},2\right]$};

\draw[rounded corners=1mm] (0,1)-(0,2)-(-1,2)-cycle;
\draw[rounded corners=1mm,rotate=15] (0,1)-(0,2)-(-1,2)-cycle;
\draw[rounded corners=1mm,rotate=30] (0,1)-(0,2)-(-1,2)-cycle;
\draw[rounded corners=1mm,rotate=45] (0,1)-(0,2)-(-1,2)-cycle;

\draw[line width=2pt,color=magenta!70!white]
  (-1,-0.5) .. controls (2,-2) and (3,-1) .. (3.8,2);
\filldraw (-1,-0.5) circle (2pt) node[left] {$\left[-1,-\frac{1}{2}\right]$};

\end{tikzpicture}
\end{document}
```

Oříznutí celého obrázku:

Osy a mřížka

Úhel α o velikosti 40^o, nakreslíme horní rameno otočené o 40^o.

Parabola

Trojúhelník se zaoblenými rohy a jeho tři pootočené kopie.

Beziérova křivka v růžové barvě.