Úvod do Maplu 7

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

Jiří Hřebíček, Jaroslav Ráček, Jan Pešl

 Brno 2002

Obsah

Kapitola 1

Základní popis systému Maple

V této kapitole se naučíte jak využívat informační technologie pro symbolické výpočty. Seznámíte se s přehledem nejznámějších programů pro symbolické výpočty a podrobně s programem Maple 7, jehož studentskou verzi budete mít k přístupnou v rámci tohoto kurzu distančního vzdělávání společně s ostatním softwarem na CD-ROM.

Dříve než se budeme věnovat informačním technologiím pro symbolické výpočty, tak si připomeňme, jak chápeme výpočty na počítačích, které vzešly ze slovesa počítat. Sloveso počítat (anglicky compute) bývá obvykle užíváno ve významu počítat s čísly, tedy provádět s nimi základní aritmetické operace sčítání, odečítání, násobení a dělení, tj. numerické výpočty. Numerický výpočet však v současné době neznamená pouze základní aritmetické operace, ale i mnohem složitější výpočty jako např. stanovení řešení soustav rovnic, numerických hodnot matematických funkcí, nalezení kořenů polynomů a vlastních čísel a vektorů matice, apod.. Základem numerických výpočtů v počítačích je, že aritmetické operace jsou vyhodnocovány nejprve obecně a teprve následně číselně. Mimoto tyto numerické výpočty nejsou ve většině případů přesné, protože se téměř vždy jedná o čísla zapsaná v počítači v pohyblivé řádové čárce, kde mantisa má omezený počet cifer. V posledních padesáti letech se však numerické výpočty na počítačích rozšířily do té míry, že pro většinu uživatelů počítačů znamenají matematické výpočty na počítačích a numerické výpočty totéž.

Matematické výpočty a matematické modelování však mají další důležitou část, kterou budeme nazývat *symbolické a algebraické výpočty*. Proto v další části této kapitoly si je podrobněji vysvětlíme.

1.1 Symbolické výpočty

Stručně můžeme symbolické a algebraické výpočty charakterizovat jako výpočty se symboly reprezentujícími matematické objekty. Tyto symboly mohou reprezentovat jednak čísla (celá, racionální, reálná a komplexní, případně i algebraická), booleovské hodnoty (pravda, nepravda, nevím) a znaky (písmena abecedy a další symboly), jednak mohou být používány pro matematické objekty (proměnné, matematické výrazy, rovnice a identity, posloupnosti, množiny, vektory, matice a tabulky, polynomy a funkce jedné a více proměnných a jejich derivace a integrály, grafy funkcí jedné a dvou proměnných a jejich animace, systémy rovnic, nerovnic a algebraické struktury jako grupy, okruhy a algebry a jejich prvky, orientované grafy, apod.).

Kromě toho přídavné jméno *symbolický* zdůrazňuje, že konečným cílem řešení matematického problému je vyjádření jeho řešení v explicitním analytickém tvaru nebo nalezení jeho symbolické aproximace (např. konečné funkční řady). Pojmem *algebraický* myslíme, že výpočty jsou prováděny přesně v souladu s pravidly algebry, namísto použití přibližné aritmetiky v pohyblivé řádové čárce, tak jak je tomu u klasických numerických výpočtů.

Příklady symbolických a algebraických výpočtů jsou například zjednodušování a úpravy matematických výrazů, analytické řešení rovnic, rozklad polynomů, derivování, integrování

funkcí a rozvoj funkcí v řady, analytické řešení obyčejných i parciálních diferenciálních a integrálních rovnic, exaktní řešení systémů rovnic i nerovností atd.

V posledních třiceti letech byl v matematice udělán velký pokrok v oblasti teoretických základů symbolických a algebraických výpočtů a algoritmů. K jejich rozvoji došlo jak na základě využití informačních technologií a prováděním matematických výpočtů a modelování na počítačích, tak využitím komunikačních technologií, zejména pak využití Internetu. To vedlo celosvětově ke vzniku nového oboru, který je označován mnoha různými jmény: *symbolické a algebraické výpočty*, *systémy symbolických výpočtů*, *operace se symboly*, *operace s výrazy* , *počítačová algebra*, atd. Bohužel termín symbolický výpočet je užíván v mnoha odlišných kontextech, jako logické programování a umělá inteligence, takže má velmi málo společného s matematickými výpočty.

Abychom se vyhnuli mylnému překladu, budeme pro tento druh výpočtů na počítačích dále užívat anglickou zkratku *CAS (Computer Algebra System)*, tj. česky *„systémy počítačové algebry"*, i když budou v zahraničním odborném tisku někdy značeny anglickou zkratku *SCS* nebo *SAC (Symbolic Computation Systems* nebo *Symbolic and Algebraic Computation)*, tj. česky *„systémy symbolických výpočtů"* nebo *„symbolické a algebraické výpočty"*.

Systémy počítačové algebry můžeme rozdělit do dvou kategorií:

a) *Systémy pro speciální účely*, které jsou vytvořeny pro řešení problémů z jednoho speciálního vědního oboru či oblasti. Mezi nejznámější systémy pro speciální účely ve fyzice náleží SCHOONSCHIP (vysokoenergetická fyzika), CAMAL (mechanika vesmírných těles), SHEEP a STENSOR (obecná relativita) a v matematice jsou to systémy CAYLEY a GAP (teorie grup), PARI (teorie čísel), CoCoA (komutativní algebry), MACAULAY (algebraická geometrie), DELiA (analýza diferenciálních rovnic) a LiE (Lieova teorie). Tyto systémy hrají důležitou roli ve svých vědních oborech a často jsou lepší a výkonnější než systémy pro obecné účely.

b) *Systémy pro obecné účely*, které lze využít pro modelování biologických systémů pro rozmanitost datových struktur matematických funkcí a které pokrývají co nejvíce různých aplikačních oblastí matematiky. Mezi nejrozšířenější systémy pro obecné účely náleží Maple (http://www.maplesoft.com), Mathematica (http://www.wri.com/), MathCad (http://www.mathsoft.com/), MuPAD (http://www.mupad.de/) a částečně i MATLAB (http://www.mathworks.com/), které umožňují vytvářet výkonné, přenosné programy, které důsledně využívají možnosti současných informačních a komunikačních technologií na různých počítačových platformách a operačních systémech, pro které byly vytvořeny. Tyto systémy pracují na širokém spektru počítačů od superpočítačů až po osobní počítače. Dále k těmto systémům náleží např. Derive (http://www.ti.com/calc/docs/derive.htm), HartMath (http://www.hartmath.com/), MathType (http://www.mathtype.com/features/), Reduce (http://www.uni-koeln.de/REDUCE/) a celá řada dalších, které je možno nalézt na webu na adrese http://www.SymbolicNet.org/systems/Systems.html.

Ačkoli mnoho standardních algebraických operací s matematickými výrazy je možno provádět jen s papírem a tužkou, tak u rozsáhlejších symbolických výpočtů začíná být nevýhodou větší délka vzorců a tím zdlouhavější práce odborníka, který výrazy upravuje. Další nevýhodou těchto operací je nutné neustále maximální soustředění, aby se dosáhlo bezchybnosti algebraických operací a tím správnosti výsledku.

Proto se stalo dlouhodobým cílem systémů počítačové algebry zautomatizovat v maximální možné míře procesy, umožňující řešit matematické problémy. Současné CAS mají sice ještě daleko k automatickým řešitelům matematických problémů a jsou zatíženy i častou

chybovostí, ale jsou využitelné jak ve základním i aplikovaném výzkumu a inženýrské praxi, tak i ve výuce matematiky, fyziky i chemie. CAS ulehčují jak přípravu výuky, tak zlepšují porozumění studentů probírané látce na přednáškách i cvičeních. Ulehčují rovněž práci zkoušejícím při zadávání a kontrole příkladů při písemných i ústních zkouškách. Hlavní jejich výhodou je jejich schopnost provádět bezchybně rozsáhlé algebraické výpočty a vizualizovat výsledky řešení matematických problémů na počítači.

Při ilustraci CAS se omezíme v dalším na program Maple 7, jehož studentská verze je součástí CD-ROM a je možno si v ní "vyzkoušet" příklady uváděné v této kapitole. Nejprve však program Maple 7 stručně popíšeme.

1.2 Charakteristika programu Maple 7

Maple je programový systém počítačové algebry vyvinutý během uplynulých dvaceti let společně na několika západních universitách, přičemž největší podíl práce vykonala skupina vědců sdružená pod názvem "Symbolic Computation Group" na universitě ve Waterloo v Kanadě a dále pak na federální technické universitě ETH Zürich ve Švýcarsku, kam část této skupiny přešla v roce 1990. Jméno Maple by mohlo být odvozeno z anglického akronyma ***Mathematics pleasure (Matematika potěšením),*** neboť Maple je skutečně příjemným prostředím pro využívání matematiky na počítači. Během posledních deseti let se Maple stal jedním z nejmodernějších a nejintenzivněji se rozvíjejících systémů počítačové algebry ve světě.

Současná verze Maple 7 umožňuje provádět jak symbolické a numerické výpočty a vytvářet grafy, tak doplňovat je vlastními texty a vytvářet tak tzv. ***hypertextové zápisníky*** (anglicky "worksheet"). Takto vytvořené zápisníky umožňuje Maple 7 ukládat do souboru na počítači ve svém speciálním mapleovském formátu MWS, nebo je volitelně exportovat do formátu Latexu, HTML, RTF a nově i MathML. Soubory ve formátu MWS umožňuje Maple 7 načítat zpět ke zpracování. To umožňuje snadnou přenositelnost maplovských zápisníků mezi nejrůznějšími počítačovými platformami a operačním systémy. Maple 7 dále umožňuje automatický převod svých příkazů a procedur do programovacích jazyků C a Fortran 77.

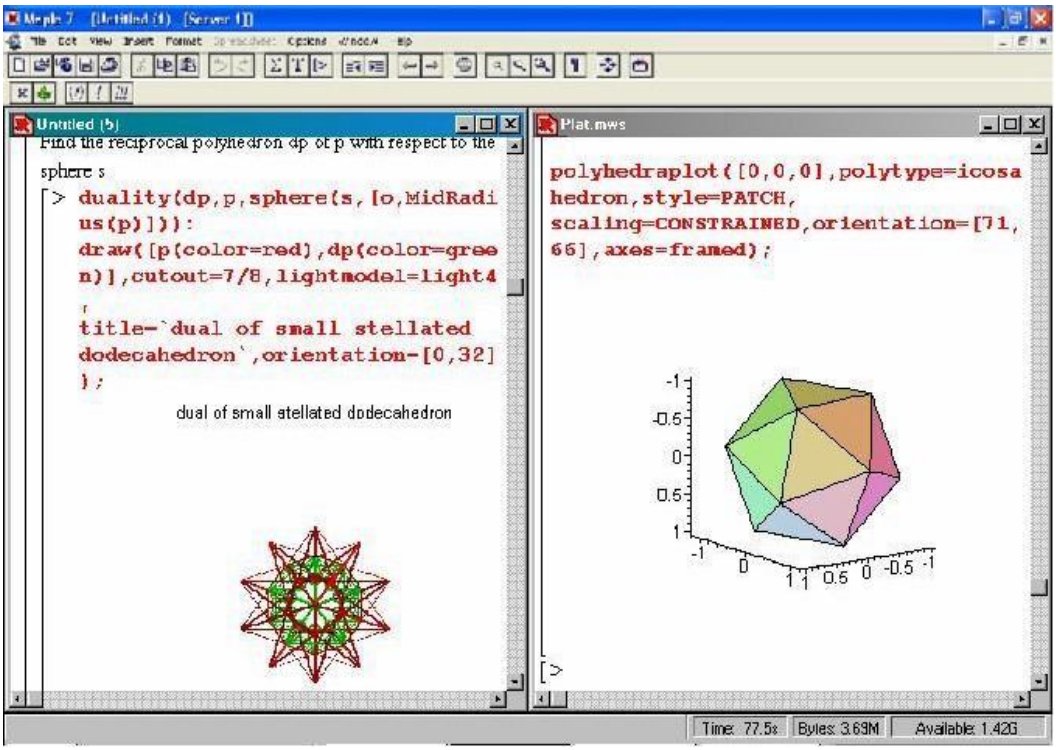
V Maplu 7 se používá vlastní programovací jazyk čtvrté generace podobný Pascalu s mnoha předdefinovanými funkcemi a procedurami. Maplovské funkce pokrývají mnoho odvětví matematiky od základů diferenciálního a integrálního počtu, lineární algebry, řešení rovnic, až k řešení diferenciálních a diferenčních rovnic, diferenciální geometrii a logice.

1.3 Uživatelské pracovní prostředí

V této sekci je popisováno uživatelské pracovní prostředí programu Maple 7, které odpovídá pracovním prostředím v operačních systémech Windows 9x, Windows NT/2000.

Systém počítačové algebry Maple 7 používá grafické uživatelské prostředí. Klíčovou částí v rámci tohoto prostředí je pracovní a komunikační rozhraní označované jako *zápisník*.

V rámci okna programu Maple 7 se standardně nachází hlavní nabídka menu a ovládací lišta reprezentovaná řadou ikon, viz obr 1-1. Pod ovládací lištou je kontextově závislá pracovní lišta s nabídkou aktuálně přístupných funkcí. Hlavní část okna programu Maple 7 je vyhrazena pro pracovní plochu, na které jsou umístěny právě zpracovávané zápisníky, při spodním okraji okna programu je stavový řádek informující o časové a paměťové náročnosti právě zpracovávaného zápisníku.



Obrázek 1-1 Okno programu Maple 7 s dvěma otevřenými zápisníky

Nabídka menu Maple 7 obsahuje vedle běžných menu pro práci se soubory (File, Edit, View, Window, Help) též některé kontextově závislé položky, jako například Style, Color, Axes, Projection, s aktuální nabídkou akcí a parametrů funkcí, použitelných v rámci aktivního objektu v zápisníku. Nejdůležitější z těchto funkcí jsou dostupné pomocí ikon na kontextově závislé pracovní liště. Většinu takto interaktivně nabízených parametrů funkcí (jako například způsob zobrazení souřadných os, barvu a tloušťku čar používaných v grafech atd.) je však možno definovat přímo jako parametry maplovských funkcí a příkazů použitých v zápisníku.

Na ovládací liště Maple 7 jsou dostupné běžné ikony usnadňující práci se zpracovávanými zápisníky, jako například otvírání, ukládání a tisk souborů zápisníku, práce se schránkou, funkce „zpět“ (undo), přepínání mezi zapisováním do zápisníku formou prostého textu nebo formou aktivních maplovských příkazů, zastavení aktuálního výpočtu (na méně výkonných počítačích velice užitečná funkce), lupa a některé další.

Kontextově závislá pracovní lišta odvíjí svoji podobu od právě aktivního objektu v zápisníku. Na obrázku 1-1 je vidět pracovní lišta s tlačítky vztahujícími se k 3D grafu funkce. Konkrétně nabízí otáčení 3D objektu kolem středu ve dvou směrech, přepínání mezi různými způsoby znázornění plochy, od vybarvené sítě až po tečkovanou interpretaci, různé způsoby znázornění souřadných os až po vynucení konstantního měřítka na všech osách. Měřítka jednotlivých os automaticky přizpůsobují tvaru a rozsahu grafu.

Pro úplné začátečníky, kteří se neorientují ve významu jednotlivých ikon, je tu možnost zapnout si interaktivní nápovědu, tzv. „baloon help“, automaticky vypisující názvy a stručnou charakteristiku funkcí spojených s danou ikonou.

1.4 Práce se zápisníky

Zápisníky jsou hlavním uživatelským pracovním prostředím Maple 7. Umožňují uživateli pohodlné zadávání vstupních dat a příkazů, zároveň též slouží k okamžité prezentaci výstupů

programu Maple 7. Po spuštění programu Maple 7 se na pracovní ploše programu automaticky otevře nový prázdný zápisník.

Práce v nového zápisníku spočívá v zapisování jednotlivý příkazů Maple 7 a vytváření základních struktur, jež jsou zápisníkem podporovány.

Těmito strukturami jsou:

- *prováděcí skupiny* (anglicky *execution groups*)
- *textové odstavce* (anglicky *paragraphs*)
- *sekce* (anglicky *sections*)
- *hypertextové odkazy* (anglicky *hyperlinks*)

1.4.1 Prováděcí skupiny

Prováděcí skupiny usnadňují práci s matematickým jádrem Maple 7. Umožňují přehledné zadávání a provádění jednotlivých příkazů a následné zobrazování výsledků.

Posloupnost příkazů (prováděcí skupina) v Maple 7 je v podstatě algoritmickým popisem řešení konkrétní matematické úlohy. Úpravami vstupních parametrů příkazů jazyka Maple 7 lze snadno získat řešení pro celou třídu daných problémů. Zápisníky umožňují jednoduše zacházet s celými prováděcími skupinami, a tak procházet celým postupem řešení.

Prováděcí skupiny jsou základní výpočetním blokem v zápisníku. Jejich primárním účelem je kombinování jednoho či více příkazů a jejich výsledků do samostatné, znovupoužitelné jednotky. Příkazy a výsledky patřící do jedné prováděcí skupiny lze snadno poznat díky veliké hranaté svorce nalevo od vstupních řádku. Jednotlivé řádky se vstupními příkazy bývají uvozeny symbolem „>“ (větší než).

■ Příklad 1.1:

Příklad prováděcí skupiny se třemi příkazy a zobrazenými výsledky.

```
> r:=8;
> Obvod:= evalf(2*Pi*r);
> Obsah:= evalf(Pi*r^2);
```

$$r := 8$$

$$Obvod := 50.26548246$$

$$Obsah := 201.0619299$$

K doplnění příkazů a výsledků může prováděcí skupina obsahovat i vysvětlující textový odstavec.

■ Příklad 1.2:

Následující prováděcí skupina obsahuje řádek doprovodného textu, jeden příkaz a výsledek.

Maple 7 příkaz a vypočtený výsledek:

```
> expand((a+b)^3);
```

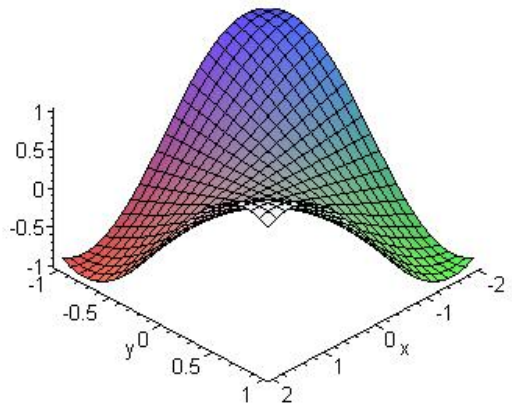
$$a^3 + 3\,a^2\,b + 3\,a\,b^2 + b^3$$

Výsledky příkazů mohou být *numerické* (Příklad 5.1), *symbolické* (Příklad 5.2) či *grafické* (viz Příklad 5.3).

■ Příklad 1.3:

Následující příkaz vykreslí třírozměrný graf funkce *sin(xy)*. V prostředí Maple 7 lze s tímto grafem pomocí ukazatele myši otáčet kolem středu.

```
> plot3d(sin(x*y) , x = -2..2, y = -1..1 );
```



Z důvodů přehlednosti v dalších příkladech již budeme vynechávat velkou hranatou svorku na levé straně, označující příkazy a jejich výsledky, patřící do jedné prováděcí skupiny. Nebude-li uvedeno jinak, budeme automaticky předpokládat, že příkazy stojí samostatně, tedy že nejsou sdruženy v žádných prováděcích skupinách.

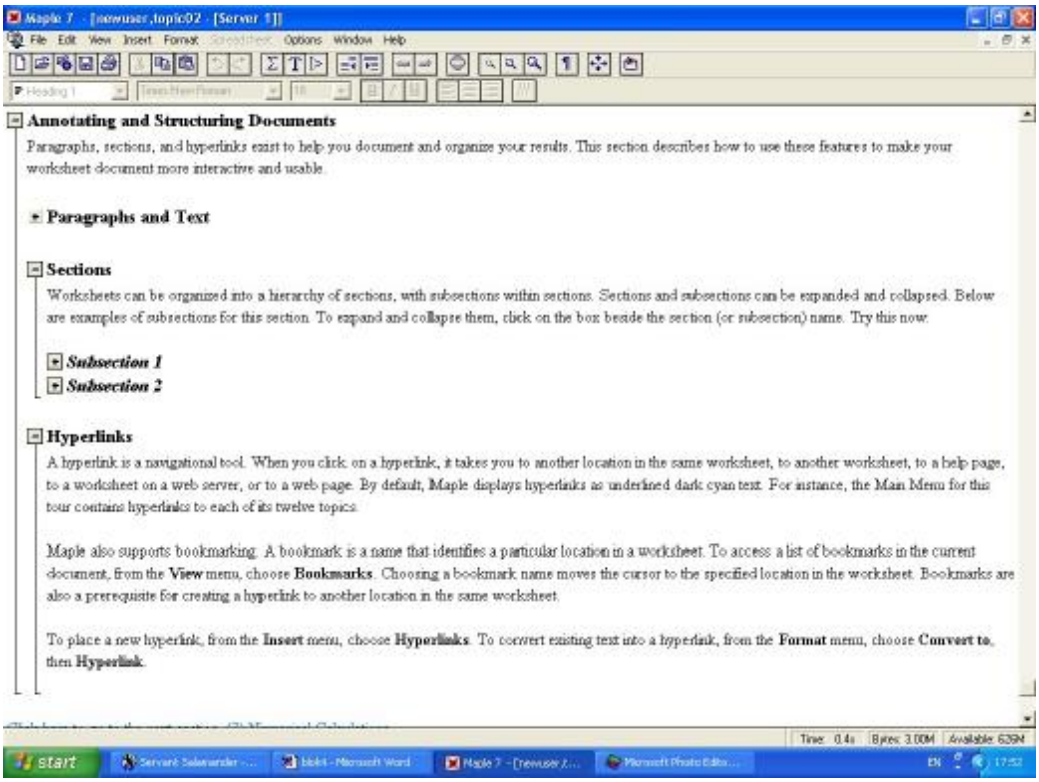
1.4.2 Textové odstavce

Odstavec prostého textu v programu Maple 7 je stejný jako text, se kterým se lze běžně setkat v textových procesorech. Odstavce mohou obsahovat text formátovaný pomocí různých stylů. Text může být obsažen i v jednotlivých prováděcích skupinách. Funguje zde i běžné zarovnávání textu na střed či okraje stránky stejně jako u textových procesorů.Všechny běžně využívané funkce pro formátování textu jsou snadno dostupné na kontextově závislé pracovní liště.

1.4.3 Sekce v zápisnících

Zápisník programu Maple 7 se tedy skládá z doprovodných textů, prováděcích skupin a vložené grafiky. Tyto součásti zápisníku mohou být organizovány v hierarchické struktuře založené na *sekcích* a *podsekcích*.

Zápisník lze pomocí *sekcí* přehledně členit. Sekce jsou části zápisníku, které lze podle potřeby jednotlivě otevírat a zase naopak zavírat.



Obrázek 1-2 Zápisník Maple 7 popisující anglicky strukturování zápisníků

K ovládání sekce slouží tlačítko vlevo od názvu sekce se symboly "+" respektive "-".

- "+" značí, že sekce je momentálně zavřená a kliknutím na tlačítko se otevře
- "-" znamená, že sekce je otevřená a po kliknutím na tlačítko se uzavře

Veškerý obsah konkrétní otevřené sekce je pro přehlednost a snazší orientaci v textu ohraničen svorkou na levé straně, podobně jako prováděcí skupiny.

Díky tomu, že podsekce lze do sebe libovolně vnořovat, jsou výborným nástrojem určeným k víceúrovňovému strukturování zápisníku. V sekcích lze například skrýt podrobný vysvětlující text, který při běžném provádění zápisníku ruší a znepřehledňuje výsledky, nebo dokonce i alternativní způsob řešení daného problému zapsaný v prováděcích skupinách.

Sekce a podsekce se vkládají do zápisníku pomocí menu `Insert/Section`, respektive `Insert/Subsection`.

1.4.4 Hypertextové odkazy

Do textu zápisníku lze vkládat i hypertextové odkazy, které v rámci Maple 7 rozlišujeme na dva druhy.

- *odkazy na zápisníky,*
  které slouží k přímému propojení jednotlivých zápisníků do ucelené struktury, po které se pak můžeme snadno pohybovat. Můžeme takto zpřístupnit i zápisníky umístěné v síti Internet a Intranet.
- *odkazy na soubory jiného typu,*
  které slouží k tomu, aby při vybrání takovéhoto odkazu je spuštěn externí prohlížeč webovských stránek, kterému je předán vybraný odkaz. Takovéto odkazy pracují buď s URL adresou daného souboru nebo je nutno jim zadat absolutní cestou v rámci lokální adresářové struktury. Bohužel při předání odkazu externímu prohlížeči dochází

často k nežádoucí interpretaci odkazu. Z toho důvodu nelze u těchto odkazů používat směřování pomocí relativních cest, což poněkud snižuje možnosti jejich využití

1.4.5 Průběh výpočtu ve vytvořeném zápisníku

Způsob práce a prohlížení již vytvořených zápisníků se odvíjí od způsobu jejich sestavení, tj. od uspořádání jejich prováděcích skupin. Pokud se umístí kurzor na libovolný řádek v prováděcí skupině a stiskne klávesa `[Enter]`, znamená to, že se všechny příkazy v dané prováděcí skupině provedou, a to v pořadí, v jakém jsou ve skupině uvedeny za sebou. Výsledky výpočtu se však zobrazí na konci prováděcí skupiny. Kurzor se poté automaticky přesune na první řádek následující prováděcí skupiny.

Prohlédnout již vytvořený zápisník lze tedy nejlépe takto:

- pomocí myši či klávesnice umístíte kurzor na první řádek první prováděcí skupiny v daném zápisníku a pak stisknete `[Enter]`,
- Maple 7 zobrazí výstupy a výsledky vykonané prováděcí skupiny,
- po prostudování výstupů pokračujte dále opět stiskem klávesy `[Enter]`, čímž se spustí a provede následující prováděcí skupina.

Tímto způsobem postupně projdete všechny prováděcí skupiny v zápisníku.

Poznámka: Mnohdy se v prováděcích skupinách na konci zápisníku pracuje s dílčími výsledky získanými během výpočtu v předchozích prováděcích skupinách. Pokud příslušné prováděcí skupiny nebyly vykonány, nejsou tyto mezivýsledky k dispozici a po spuštění prováděcí skupiny uvnitř zápisníku se může objevit chybové hlášení. Prováděcí skupiny je tedy třeba volat (vykonávat) v závislosti na daném algoritmu. Obvykle popořadě tak jak jsou v zápisníku postupně definovány.

1.5 Základní příkazy a operace

Syntaxe používaná programem Maple 7 pro zapisování příkazů je podobná syntaxi programovacích jazyků Pascal a C. Příklady použité v této kapitole i všechny ostatní příklady jsou zobrazeny tak, jak se zpravidla jeví při skutečné práci s programem Maple 7.

1.5.1 Způsob zápisu příkazů v zápisnících

Nejprve uvedeme několik obecných informací o způsobu zapisování příkazů na řádky zápisníku.

- Každý příkaz v zápisníku, obsahující příkaz Maple 7, musí být ukončen středníkem „ ; “ nebo dvojtečkou „ : “.
- Po stisknutí klávesy `[Enter]` je celá aktuální prováděcí skupina předána jako vstup "výpočetnímu jádru" programu Maple 7, které ji zpracuje.

Pokud je řádek se zpracovávaným mapleovským příkazem zakončen středníkem, znamená to, že výsledek provedené operace se zobrazí na dalším řádku. Je-li řádek zakončen dvojtečkou, Maple 7 vyhodnotí zadaný příkaz, ale nic nezobrazí a očekává další příkaz. Tohoto se využívá zejména k potlačení tisku mezivýsledků v průběhu delších výpočtů.

Při zápisu posloupnosti příkazů či dlouhých algebraických výrazů je zapotřebí mít možnost přecházet na další řádek bez toho, že by se prozatím napsaný kód nějakým způsobem

vyhodnocoval, jak se v zápisníku děje po prostém stisku klávesy [Enter]. To je možné provádět následujícím způsobem:

- Posunout kurzor na nový řádek, bez vyhodnocení dosud napsaného kódu, nám v zápisnících umožňuje kombinace kláves [Shift] + [Enter].

1.5.2 Aritmetické operace a přiřazení hodnoty proměnné

Následující příklad ukazuje, jak pracovat v Maple 7 s přiřazením hodnoty maplovské proměnné a běžnými aritmetickými operacemi.

■ **Příklad 1.4:**

```
> a:= 5/2;
```

$$a := \frac{5}{2}$$

```
> b:=6!;
```

$$b := 720$$

```
> c:= a*b;
```

$$c := 1800$$

```
> d:= c^2+Pi*(a-e);
```

$$d := 3240000 + \pi\left(\frac{5}{2} - e\right)$$

```
> e:= c-12345;
```

$$e := -10545$$

```
> f:=d*(b-e^a);
```

$$f := \left(3240000 + \frac{21095}{2}\pi\right)(720 - 111197025\sqrt{-10545})$$

Z výše uvedeného příkladu je vidět, že Maple 7 zobrazuje výsledky v „matematické notaci“ a pokud možno přesně, tj. v tvaru celých čísel, zlomků, případně výrazů, kde se vyskytující konstanty (π a komplexní jednotka I) i odmocniny.

Pokud chceme mít výsledek uvedený pouze přibližně, např. jako desetinné číslo, tak použijeme maplovskou funkci evalf(). Maple 7 zobrazuje reálná čísla v pohyblivé řádové čárce s mantisou standardně na 10 desetinných míst.

```
> evalf(f);
```

$$.2356657883\ 10^{10} - .3737494002\ 10^{17}\ I$$

V případě, že požadujeme větší počet desetinných míst, uvedeme jejich počet jako další parametr ve funkci evalf(). Počet míst mantisy není omezen, ale prodloužíme tím délku výpočtu.

```
> evalf(f,20);
```

$$.23566578829298916078\ 10^{10} - .37374939999854099987\ 10^{17}\ I$$

Následující tabulka podává stručný přehled aritmetických operátorů, základních matematických funkcí a konstant definovaných v Maple 7. Zároveň ukazuje způsob jejich zápisu v prostředí zápisníku.

Zápis v Maple 7	Význam	Matematický zápis
`x + y`	Součet	$x + y$
`x - y`	Rozdíl	$x - y$
`x * y`	Součin	$x * y$
`x / y`	Podíl	$\frac{x}{y}$
x^y nebo `x**y`	Umocňování	x^y
`sqrt(x)` nebo x^(1/2)	druhá odmocnina	$\sqrt{x}$
`x!`	Faktoriál	$x!$
`abs(x)`	absolutní hodnota	$\lvert x \rvert$
`I` nebo `sqrt(-1)`	komplexní jednotka	i nebo $\sqrt{-1}$
`Pi`	Ludolfova konstanta	π
`infinity`	symbol pro nekonečno	∞

Tabulka 1-1 Základní operátory, funkce a konstanty

Z příkladu 1.4 je vidět, že v prostředí zápisníku můžeme definovat proměnné. Proměnnou definujeme prostým přiřazením hodnoty jménu proměnné nebo jejím zápisem ve výrazu.

- přiřazovací operátor má podobu „ := “ (samotné „ = “ má jiný význam).

Jméno maplovské proměnné musí splňovat následující podmínky:

- musí začínat písmenem, ať už malým či velkým (POZOR! Maple 7 rozlišuje mezi malými a velkými písmeny)
- může být složeno z následujících znaků:
  - písmena,
  - číslice,
  - znaku podtržítko „ _ “ ,
- maximální délka jména proměnné je závislá na počítačové platformě, na 32-bitových systémech (případ Windows 9x) je to 524 271 znaků, na systémech 64-bitových je maximální délka jména proměnné 34 359 738 335 znaků.
- jako jméno proměnné nemůžeme definovat žádné z klíčových slov, která uvedeme dále.

Poznámka: Znak podtržítko „_“ na začátku jména proměnné je vyhrazen pro globální systémové proměnné, proto se doporučuje taková jména nepoužívat.

■ **Příklad 1.5:**
Chceme-li tedy například přiřadit hodnotu 77, proměnné `polomer` zapíšeme v Maple 7 příkaz

```
> polomer := 77:
```

Poznámka: Maple 7 na platformách s Windows 9x sice podporuje kódování češtiny, nicméně její používání ve jménech proměnných rozhodně nedoporučujeme, neboť to může vést k neočekávané interpretaci zadaných jmen.

Vzhledem k tomu, že jména proměnných, jakož i funkcí a procedur, jsou programem Maple 7 rozlišována podle malých a velkých písmen, tak po zadání následujících příkazů obdržíme:

```
> Polomer := 50:
> polomer – Polomer;
```

$$27$$

Proměnné může být přiřazen jakýkoli výraz či maplovská struktura. Jako příklad uveďme tři následující přiřazení:

■ **Příklad 1.6:**
Přiřazení rovnice:

```
> rovnice := 3 * x^2 - x + 11 = 0;
```

$$rovnice := 3x^2 - x + 11 = 0$$

Přiřazení struktury definující graf funkce *2*cos(x):*

```
> graf := plot (2*cos(x), x = -Pi .. Pi):
```

Strukturu definující graf funkce zde z úsporných důvodů nevypisujeme.

Pro úplnost ještě uveďme, že v Maple 7 je možno definovat jako jméno proměnné dokonce posloupnost znaků obsahující mezery. Takové jméno proměnné se uzavře do jednoduchých uvozovek a dále se s ním pracuje běžným způsobem.

```
> `Toto je jmeno promenne` := 23;
```

$$Toto\ je\ jmeno\ promenne := 23$$

Poznámka: Při výpisu výsledků však Maple 7 jednoduché uvozovky uzavírající jméno proměnné vynechává. Výstupy z prováděcích skupin, ve kterých jsou použita jména proměnných s mezerami, jsou pak zpravidla velice těžce čitelné, proto je nedoporučujeme používat. Uzavřením jména proměnné do jednoduchých uvozovek se význam jména nemění.

1.5.3 Klíčová slova v programu Maple 7 :

Klíčová slova jsou určena k definování programových struktur při programováni v Maple 7. Tato slova jsou vyjmuta a nelze je používat jako jména proměnných.

```
and   by   do        done   elif
else  end  fi        for    from
if    in   intersect local  minus
mod   not  od        option options
or    proc quit      read   save
stop  then to        union  while
```

Kromě této nevelké skupiny klíčových slov však v Maple 7 existují stovky slov, jež jsou zařazeny do typu `protected`. Tento typ slouží jako ochrana vyhrazených slov před náhodným předefinováním jejich významu. Patří sem slova jako `sin, cos, ln, exp, Pi` a mnohá další. Slova typu `protected` nemohou být použita jako jména proměnných, při pokusu přiřadit jim nějakou hodnotu akce skončí chybovým hlášením:

```
> Pi := 1;
```

```
Error, attempting to assign to `Pi` which is protected
```

V případě potřeby můžeme jakékoli slovo typu `protected` z tohoto typu vyjmout a poté předefinovat. K tomu slouží funkce `unprotect()`.

```
> unprotect(Pi): Pi := 1;
```

$$\pi := 1$$

A naopak, jméno libovolné proměnné lze zařadit do typu `protected`, takže v průběhu další práce s Maple 7je zaručeno, že jeho hodnota nebude náhodně či omylem změněna. K tomuto úkolu složí mapleovská funkce `protect()`.

Seznam všech slov zařazených v typu `protected` obdržíte po zadání příkazu:

```
> select(type,{unames(),anames(anything)},protected);
```

Vzhledem k jejich počtu je zde nebudeme vypisovat.

1.5.4 Odkazování na předchozí výsledky

Každý příkaz zadaný a zpracovaný programem Maple 7v rámci zápisníku má svoji hodnotu, kterou získá během svého vyhodnocení. Na hodnotu již vyhodnocených příkazů se můžeme přímo odkazovat pomocí speciální systémové proměnné „ % “. To znamená, že není vždy nutné přiřazovat výsledek příkazu do nějaké proměnné. To je velmi užitečné, zajímá-li nás z celého průběhu výpočtu, rozloženého do více kroků, pouze konečný výsledek.

U složitějších úloh bychom však určitě nevystačili pouze s odkazem na bezprostředně předcházející výsledek. Maple 7 však umožňuje odkazovat se zpětně až na třetí předchozí vyhodnocený výsledek.

Symbol: % - odkazuje na poslední vyhodnocený výsledek
%% - odkazuje na předposlední vyhodnocený výsledek
%%% - odkazuje na třetí vyhodnocený výsledek ve zpětném pořadí

■ Příklad 1.7:
Využití symbolu % ilustrujme na následujících příkazech.

```
> (5 + 2)^2;
```

$$49$$

```
> % - 45;
```

$$4$$

```
> 1 + %%;
```

$$50;$$

```
> (%%% / 7) * 3;
```

$$21$$

```
> sqrt(%%%);
```

$$2$$

Z výše uvedeného příkladu je vidět, že zjednodušení práce může být výrazné. Přesto doporučujeme přiřadit výsledek příkazu do proměnné pro lepší přehled v zápisníku.

1.6 Interaktivní nápověda

Interaktivní nápověda v Maple 7 slouží ke snadné a rychlé orientaci ve stovkách maplovských příkazů a funkcí a jejich parametrů. V Maple 7 je nápověda řešena jako systém textových dokumentů propojených hypertextovými odkazy. Každá standardní funkce Maple 7 má zpracovánu vlastní stránku s nápovědou. Jednotlivé stránky nápovědy mají pevnou strukturu, skládající se z následujících po sobě jdoucích částí:

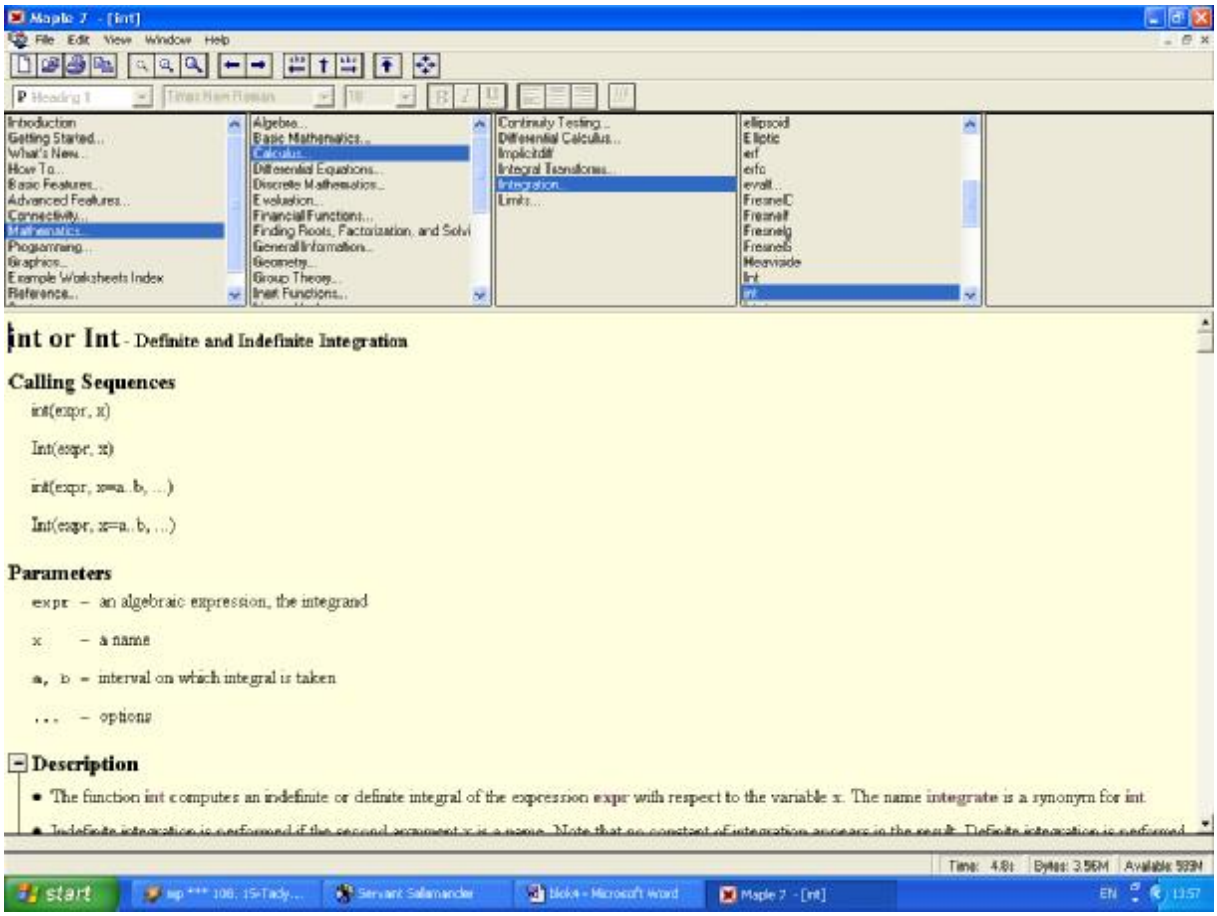
- jméno a charakteristika funkce
- popis volání funkce
- definice parametrů funkce
- podrobný popis vlastností funkce
- příklady použití funkce
- sbírka hypertextových odkazů na příbuzná témata v nápovědě Maple 7

Systém interaktivní nápovědy v Maple 7 spočívá v tom, že umožňuje zpřístupnit přímo stránku týkající se zadané funkce. Ukážeme to na příkladu.

■ Příklad 1.8:
Potřebujeme-li například nápovědu k funkci `int` napišme na řádek zápisníku příkaz

```
> ?int
```

a stiskněme klávesu [Enter] . Tím zobrazíte požadovanou stránku nápovědy.



Obrázek 1-3: Nápověda k funkci `int`

Je-li již jméno funkce `int` v zápisníku použito, tak stačí na něj přenést kurzor, myší či klávesovými šipkami, a poté vybrat položku `` `Help on int` `` z menu `Help` nebo jen stisknout klávesy `Ctrl + F1`.

Pokud jsme v situaci, kdy hledáte funkci, jejíž jméno neznáte, nezbývá vám, než procházet systém stránek nápovědy manuálně. Systém stránek nápovědy je však hierarchicky členěn a díky tomu se v něm dá velice dobře vyhledávat. Otevřít systém nápovědy můžeme například pomocí položky `Using Help` v menu `Help` .

1.7 Knihovny funkcí

Maple 7 při řešení úloh umožňuje použít obrovské množství funkcí. Ve standardní instalaci současné verze Maple 7 je dostupných více než 3000 funkcí. Funkce jsou uloženy v takzvané *knihovně funkcí.*

Z důvodu zpřehlednění práce se stovkami přístupných funkcí jsou funkce v rámci knihovny rozděleny do takzvaných *balíků* (packages).

Kromě standardní knihovny funkcí je možno s programem Maple 7 využívat též tzv. s*hare library*. Tato knihovna je sestavena z funkcí a balíků napsaných uživateli Maple 7 a obsahuje mnoho funkcí použitelných přímo v praxi. Tato knihovna je nyní uložena na webovském serveru Maple na adrese http://www.maplesoft.com v tzv. Maple Application Center.

1.7.1 Standardní knihovní funkce – Standard library functions

Funkce z tohoto balíku jsou automaticky přístupné a je možné je volat jejich jménem ihned po startu programu. Není nutné je inicializovat jako funkce z ostatních balíků, viz 1.7.2. Sem patří hlavně běžné matematické funkce a pak také příkazy Maple 7, umožňující manipulaci a vyhodnocování zpracovávaných výrazů. Seznam standardních knihovních funkcí je dostupný příkazem:

```
> ?index[function];
```

1.7.2 Balíky knihovních funkcí – Library packages

Balíky funkcí jsou jakési skupiny funkcí, definovaných v rámci knihovny funkcí Maple 7. Funkce vztahující se k řešení určité třídy matematických úloh jsou společně zařazeny do jednoho balíku. Potřebujeme-li například pracovat s objekty třírozměrného euklidovského prostoru, najdeme všechny funkce vztahující se k této problematice v jednom balíku, konkrétně zde jde o balík `geom3d`.

Pokud chceme používat funkce definované jako součást některého balíku, existují dva způsoby jejich volání:

a) volání pomocí *dlouhých jmen* (`long names`)
b) volání pomocí *krátkých jmen*

Volání pomocí dlouhých jmén funkcí

Dlouhá jména funkcí jsou vždy platná a není třeba je žádným způsobem inicializovat. Dlouhé jméno funkce je určeno *jménem balíku,* do kterého je funkce zařazena a *jménem funkce* v rámci balíku.

Chceme-li například použít funkci `point` z balíku `geom3d` je způsob jejího volání následující: `geom3d[point].`

Tato metoda rozdělení funkcí vylučuje kolizi jmen funkcí z různých balíků. Možnost nechtěného předefinování významu některé z funkcí během práce je v kterémkoli programu velice nebezpečná. Uvědomíme-li si navíc, že v programu Maple 7 můžeme používat tisíce různých funkcí, vidíme, že opatření zabraňující takovéto chybě jsou zcela nezbytná.

Volání pomocí krátkých jmen funkcí

Výše uvedené konstrukci volání funkcí pomocí dlouhých jmen se můžeme vyhnout pomocí *inicializace zkráceného volání funkcí.* Tato možnost slouží hlavně k zjednodušení zapisovaného kódu a usnadňuje též manipulaci s funkcemi. K inicializaci krátkých jmen funkcí z daného balíku slouží příkaz `with();`

Pokud zadáme jako parametr příkazu pouze jméno balíku: `with(jméno_balíku);` inicializujeme současně všechny funkce z daného balíku.

Pokud jako druhý parametr specifikujeme konkrétní jméno funkce, inicializuje se jen uvedená funkce: `with(jméno_balíku, jméno_funkce);`.

Po zavedení zkráceného volání funkcí z určitého knihovního balíku jsou vypsána jména nově inicializovaných funkcí, pokud došlo ke shodě jmen některé inicializované funkce s již definovaným jménem funkce či proměnné, platí význam nově inicializované funkce!

Zkrácené volání funkcí z balíku `geom3d` tedy inicializujeme příkazem:

```
> with(geom3d);
```

Seznam všech dostupných knihovních balíků i s jejich stručným popisem je dostupný pomocí příkazu:

```
> ?index[packages];
```

1.7.3 Načtení funkcí z knihovny – příkaz `readlib()`

Mnoho funkcí z knihovny funkcí je načteno ihned při startu programu Maple 7. Pokud však chceme použít funkci, která není při startu definována automaticky, musíme ji nejdříve z knihovny funkcí načíst a tím ji definovat. K tomuto používáme příkaz `readlib()`.

Informaci o tom, zda je nutno konkrétní funkci načítat z knihovny funkcí pomocí příkazu `readlib()` najdeme na stránce nápovědy dané funkce.

Parametrem příkazu `readlib()` je pouze jméno funkce, které si přejeme načíst z knihovny funkcí Maple 7.

■ **Příklad 1.9:**
Potřebujeme-li používat funkci `log10()`, vracející hodnotu dekadického logaritmu, tak pomocí příkazu `readlib()`ji zpřístupníme pomocí příkazu.

```
> readlib(log10);
```

proc(x) ... end

1.8 Matematické funkce

V této části je uveden výčet názvů základních matematických funkcí, jež jsou programem Maple 7 podporovány.

1.8.1 Trigonometrické a hyperbolické funkce:

Trigonometrické a hyperbolické funkce patří mezi standardní knihovní funkce a nemusí se před svým voláním nijak inicializovat.

`sin()`	`arcsin()`	`sinh()`	`arcsinh()`
`cos()`	`arccos()`	`cosh()`	`arccosh()`
`tan()`	`arctan()`	`tanh()`	`arctanh()`
`cot()`	`arccot()`	`coth()`	`arccoth()`

1.8.2 Logaritmy a exponenciální funkce:

`ln(), log()`	přirozený logaritmus, logaritmus o základu **e**
`log[b]()`	logaritmus o základu b
`log10()`	dekadický logaritmus, je třeba definovat pomocí `readlib(log10);`
`exp()`	exponenciální funkce, hodnota exponenciální funkce o základu **e**

1.8.3 Celočíselné funkce:

`factorial(),!`	funkce faktoriál, symbol ! se používá v běžné formě postfixové notace
`binomial()`	binomický koeficient, volání `binomial(n,k)` kde $0 \le k \le n$, definován vztahem $\frac{n!}{k!(n-k)!}$

Kapitola 2

Úpravy matematických výrazů

Při řešení příkladů se složitými matematickými výrazy se můžeme snadno dostat do situace, kdy při „ručních“ úpravách těchto výrazů uděláme chybu nebo si nejsme zcela jisti správností dosaženého výsledku. Právě zde se nabízí možnost použití CAS s jejichž pomocí můžeme algebraický výraz převést do požadovaného tvaru, případně si ověřit již dosažený výsledek. Tím se ušetří velké množství práce a času.

Jednou z velkých předností CAS je jeho schopnost pracovat se symbolickými algebraickými výrazy tak, jak jsme to viděli v předchozí kapitole. Tato vlastnost se možná trochu paradoxně projevuje právě tím, že většina CAS do vyjádření zadaného algebraického výrazu zasahuje zcela minimálně. Kromě numerických výpočtů číselných aritmetických výrazů a zjednodušování zlomků zde neexistuje prakticky žádné automatické upravování algebraických výrazů do nějakého předepsaného tvaru. CAS ponechávají zcela na uživateli, jakým způsobem bude daný výraz upraven.

Jednotlivé systémy počítačové algebry nabízí velké množství příkazů a funkcí pro úpravy algebraických výrazů. Podle toho, na jaký tvar chceme upravovaný výraz převést, volíme jednotlivé funkce a jejich parametry.

V následující kapitole se zaměříme na objasnění základních funkcí pro úpravu matematických výrazů v Maple 7, které jsou:

- `simplify`
- `expand`
- `factor`
- `normal`
- `convert`

2.1 Zjednodušování algebraických výrazů - funkce `simplify`

Funkce `simplify()` je nejvíce používaná funkce v Maple 7 k aplikaci celé řady zjednodušovacích pravidel, které jsou vhodné ke zpracování zadaného algebraického výrazu. Při volání funkce `simplify()` je zadaný algebraický výraz prohledán, jsou v něm identifikována volání funkcí, mocniny a odmocniny, a. poté jsou provedena všechna zjednodušení platná pro daný výraz.

Při volání funkce `simplify()` můžeme pomocí druhého parametru specifikovat skupinu zjednodušujících matematických pravidel, jež mají být při zjednodušování výrazu použita. Zjednodušovací pravidla z ostatních skupin pak při tomto zjednodušení použita nebudou. Můžeme tak určit, které matematické funkce v zadaném výrazu zjednodušeny budou a které naopak zjednodušeny nebudou.

Jako druhý parametr funkce `simplify()` lze volit tato jména skupin zjednodušujících pravidel:

- `trig` – tento parametr použijeme, pokud chceme ve výrazu zjednodušit obsažené trigonometrické funkce.
- `radical` – parametr se používá pro úpravu výrazů s racionálními mocninami.
- `power` – parametr použijeme v případě, kdy chceme zjednodušit mocniny a logaritmy
- `sqrt` – jedná se o parametr pro úpravu výrazů, v nichž se vyskytuje čtverec mocnin.
- `ln` - parametr je vhodný pro zjednodušování výrazů s logaritmy.
- `Ei, GAMMA, RootOf, atsign, hypergeom, polar` – tyto parametry jsou méně obvyklé a nebudeme je již podrobněji popisovat, neboť jejich popis můžeme získat pomocí nápovědy.

■ **Příklad 2.1:**

Nejprve ukážeme použití příkazu `simplify` pouze s jedním základním parametrem.

```
> 4^(1/2)+3;
```

$$\sqrt{4}+3$$

```
> simplify(%);
```

$$5$$

```
> ((a^4-b^4)/(a^2*b^2))/((1+b^2/a^2)*(1-2*a/b+a^2/(b^2)));
```

$$\frac{a^4-b^4}{a^2\,b^2\left(1+\frac{b^2}{a^2}\right)\left(1-\frac{2\,a}{b}+\frac{a^2}{b^2}\right)}$$

```
> simplify(%);
```

$$\frac{a+b}{-b+a}$$

Jak je vidět, lze příkaz `simplify` použít jak pro výpočet hodnoty číselného výrazu, tak i pro zjednodušení výrazu s proměnnými. Nyní ukážeme na dvou případech použití příkazu `simplify` s parametrem `trig`:

```
> sin(x)^2+cos(x)^2;
```

$$\sin(x)^2+\cos(x)^2$$

```
> simplify(%,trig);
```

$$1$$

```
> c:= exp(ln(cos(2*x)+sin(x)^2) + 1);
```

$$c := \mathbf{e}^{(\ln(\cos(2\,x)+\sin(x)^2)+1)}$$

```
> simplify(c);
```

$$\cos(x)^2\,\mathbf{e}$$

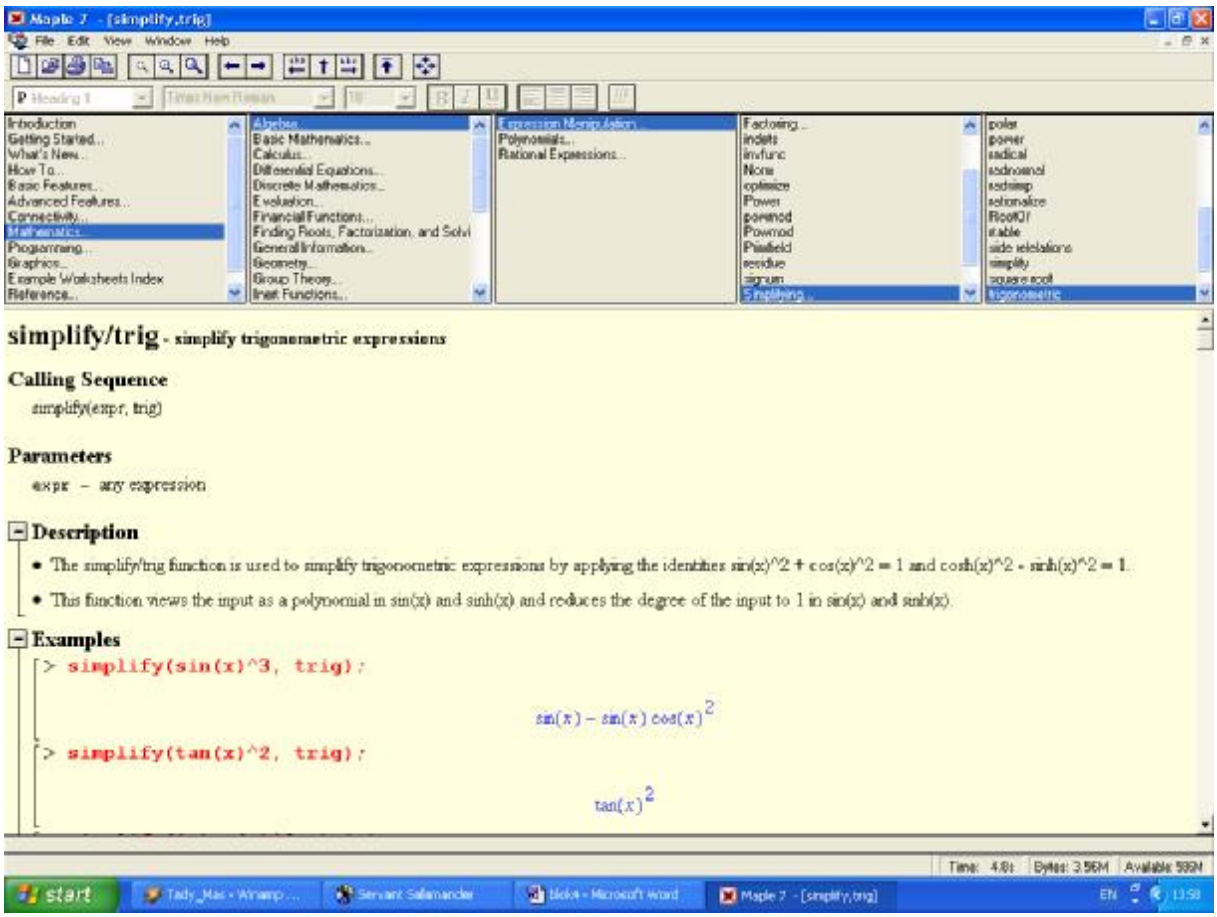
```
> simplify(c,trig);
```

$$\mathbf{e}^{(\ln(\cos(x)^2)+1)}$$

Vidíme, že použití parametru `trig` vede ve druhém případě k složitějšímu výrazu, neboť se nepoužila skupina zjednodušujících pravidel pro funkci `ln`.

Poznámka: Pomocí příkazu `?simplify[jméno_skupiny_pravidel]` lze vypsat nápovědu ke každé skupině zjednodušujících pravidel. V nápovědě ke každé ze skupin jsou definována skupinou používaná zjednodušující pravidla. Samotné jméno skupiny vesměs odpovídá typu matematických funkcí, které lze s její pomocí zjednodušovat.

Například pomocí příkazu `?simplify[trig]` se vypíše nápověda `pro` skupinu umožňující zjednodušování trigonometrických funkcí, viz obr. 2-1.



Obrázek 2-1: Nápověda pro skupinu `trig`

2.2 Rozvoj algebraických výrazů - funkce `expand`

Úlohou funkce `expand` je rozvinout algebraický výraz do řady součtů jeho podvýrazů. Rozvíjí se takové algebraické výrazy, u kterých je to možné, do běžné podoby polynomu. Mimo to funkce `expand` umí též pracovat s většinou běžných matematických funkcí jako jsou například funkce: *sin, cos, tg, ln, exp* apod. Dovede také do podoby součtu rozvinout výrazy obsahující zmíněné matematické funkce.

Stejně jako funkci `simplify` lze volat funkci `expand` s jedním nebo více parametry. Prvním a zároveň jediným povinným parametrem funkce je výraz, který má být rozkládán. Jako další parametry lze uvést funkce z výrazu, které nemají být dále rozkládány.

■ **Příklad 2.2:**

V následujících příkazech jsou ukázány možnosti funkce `expand` s jedním nebo více parametry:

```
> (x+1)*(x+2)=expand((x+1)*(x+2));
```

$$(x+1)\,(x+2) = x^2 + 3\,x + 2$$

```
> sin(x+y)=expand(sin(x+y));
```

$$\sin(x+y) = \sin(x)\cos(y) + \cos(x)\sin(y)$$

```
> a:=(x+1)*(y+z);
```

$$a := (x+1)\,(y+z)$$

```
> expand(a);
```

$$x\,y + x\,z + y + z$$

```
> expand(a,x+1);
```

$$(x+1)\,y + (x+1)\,z$$

```
> expand(cos(2*x));
```

$$2\cos(x)^2 - 1$$

```
> expand((1+x)*(1-x)*(x-2)^2);
```

$$-3\,x^2 - 4\,x + 4 - x^4 + 4\,x^3$$

Z posledního uvedeného příkazu je vidět, že Maple 7 netřídí členy polynomu podle jejich mocnin. Pokud se nám nelíbí výsledek v podobě polynomu se nesetříděnými mocninami, tak jej můžeme setřídit pomocí funkce `sort(),` a upravit tak pořadí členů polynomu.

```
> sort(%);
```

$$-x^4 + 4\,x^3 - 3\,x^2 - 4\,x + 4$$

S funkcí `expand` jsou úzce spjaty funkce `expandon` respektive `expandoff`. Pomocí funkcí `expandon` respektive `expandoff` můžeme předem označit funkce, které mají respektive nemají být rozloženy pomocí funkce `expand`. Pro názornost a snazší pochopení použití těchto funkcí je uveden následující příklad.

■ **Příklad 2.3:**

Rozdíl mezi použití funkce `expandon` a `expandoff` ukážeme na příkladu exponenciální funkce:

```
> expand(expandoff()); expandoff(exp);
```

expandoff( )

```
> expand(exp(a+b));
```

$$\mathbf{e}^{(a+b)}$$

Vidíme, že vypnutím rozkladu pro exponenciální funkci se výše uvedený výraz nerozložil na součin dvou exponenciálních funkcí.

```
> expand(expandon()); expandon(exp);
```

expandon( )

```
> expand(exp(c+d));
```

$$\mathbf{e}^{c}\,\mathbf{e}^{d}$$

Naopak, když znovu nastavíme používání pravidel pro exponenciální funkci, tak se výše uvedený výraz rozloží na součin dvou exponenciálních funkcí.

2.3 Rozklad polynomu - funkce `factor`

Funkce `factor` rozkládá polynomy o více neznámých s celočíselnými, reálnými nebo komplexními koeficienty na součin ireducibilních polynomů. Funkce se volá s jedním nebo dvěma parametry. Jako první (povinný) parametr se uvádí výraz (polynom), který se má upravit, druhým parametrem je možné specifikovat číselnou množinu, nad kterou chceme výpočet provést. Pokud je jako první parametr funkce `factor` zadána množina nebo seznam výrazů, jsou postupně rozkládány všechny prvky této množiny.

■ Příklad 2.4:

V tomto příkladu uvedeme obvyklé způsoby použití funkcí `factor` a `ifactor`:

```
> factor(6*x^2+18*x-24);
```

$$6\,(x+4)\,(x-1)$$

```
> factor(x^5-y^5);
```

$$(x-y)\,(x^4+x^3\,y+x^2\,y^2+x\,y^3+y^4)$$

```
> a^4-2=factor(a^4-2,sqrt(2));
```

$$a^4-2=(a^2-\sqrt{2})\,(a^2+\sqrt{2})$$

```
> seznam_vyrazu:=[x^2-16,x^4-16];
```

$$seznam\_vyrazu := [x^2-16,\ x^4-16]$$

```
> factor(seznam_vyrazu);
```

$$[(x-4)\,(x+4),\ (x-2)\,(x+2)\,(x^2+4)]$$

V případě, že nejde polynom rozložit v oboru celých čísel, je možno jej rozložit v oboru reálných nebo komplexních čísel.

```
> factor(x^3+5);
```

$$x^3+5$$

```
> factor(x^3+5.);
```

$$(x+1.709975947)\,(x^2-1.709975947\,x+2.924017740)$$

```
> factor(x^3+5,complex);
```

$$(x+1,709975947)(x-0,8549879733+1,480882610I)(x-0,8549879733-1,480882610I)$$

S rozklady polynomů na ireducibilní členy souvisí také problematika rozkladů celých čísel na součin prvočísel. To se provádí pomocí funkce `ifactor`, neboť funkce `factor` rozklad na součin prvočísel neprovede.

```
> factor(420);
```

$$420$$

```
> ifactor(420);
```

$$(2)^2\ (3)\ (5)\ (7)$$

2.4 Zjednodušování racionálních výrazů - funkce `normal`

Funkce `normal` je vhodná pro zjednodušování racionálních výrazů obsahujících součty, součiny a celočíselné mocniny celých čísel a proměnných. Tyto výrazy jsou pak pomocí funkce `normal` převáděny na takzvanou *„normalizovanou formu“*, což je výraz v podobě zlomku, jehož čitatel i jmenovatel jsou pokud možno polynomy.

Funkce `normal` může být volána s jedním nebo dvěma parametry. První parametr je povinný a obsahuje výraz na který má být funkce použita. Jako druhý parametr může být použito klíčové slovo `expanded`, kterým určujeme, zda polynomiální výrazy obsažené ve výsledku budou roznásobeny.

■ Příklad 2.5:

V tomto příkladu ukážeme použití funkce `normal` jak s parametrem `expanded`, tak i bez něj:

```
> normal(x^2-(x+1)*(x-1)-1);
```

$$0$$

```
> (x^2-y^2)/(x-y)^3 = normal((x^2-y^2)/(x-y)^3);
```

$$\frac{x^2-y^2}{(x-y)^3} = \frac{x+y}{(x-y)^2}$$

```
> v:=1/x+x/(x+1);
```

$$v := \frac{1}{x} + \frac{x}{x+1}$$

```
> normal(v);
```

$$\frac{x+1+x^2}{x\,(x+1)}$$

```
> normal(v,expanded);
```

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

```
> sort(%);
```

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

Poznámka: Z předchozího příkladu je vidět, že funkci sort můžeme použít setřídění mocnin v čitateli a jmenovateli i u podílu dvou polynomů.

2.5 Zjednodušování racionálních výrazů s odmocninami - funkce `rationalize`

Funkce `rationalize` je určena pro úpravy výrazů obsahujících zlomky s odmocninami ve jmenovateli. Funkce převádí tyto zlomky na tvar, který již odmocniny ve jmenovateli nemá. Vyskytuje-li se odmocnina ve jmenovateli zlomku jako parametr jiné funkce (např. sinus), pak funkce `rationalize` tuto odmocninu neodstraní.

■ **Příklad 2.6:**

Použití funkce `rationalize` ukážeme na dvou příkladech.

```
> 2/(2-sqrt(2))=rationalize(2/(2-sqrt(2)));
```

$$2\,\frac{1}{2-\sqrt{2}} = 2+\sqrt{2}$$

```
> (1+2^(1/3))/(1-2^(1/3));
```

$$\frac{1+2^{(1/3)}}{1-2^{(1/3)}}$$

```
> rationalize(%);
```

$$-(1+2^{(1/3)})\,(1+2^{(1/3)}+2^{(2/3)})$$

2.6 Převedení více výrazů do jednoho výrazu - funkce `combine`

Funkce `combine` se používá v případech, kdy chceme výraz o více členech obsažených v součtech, součinech a mocninách převést na výraz o jediném členu. Tuto funkci lze také aplikovat na seznamy a množiny výrazů.

Funkce může být volána s jedním nebo více parametry, kde první parametr je opět povinný a obsahuje upravovaný výraz. Ostatní parametry mohou upřesňovat způsob, jakým má být transformace provedena. Těmito parametry mohou být například `trig`, `exp`, `ln`, `power` nebo `abs`. Pomocí těchto parametrů specifikujeme typy matematických funkcí, na něž má být funkce `combine` aplikována. Použití a význam parametrů je obdobný jako v případě funkce `simplify`.

■ **Příklad 2.7:**

V tomto příkladu jsou ukázány základní způsoby použití funkce `combine`:

```
> 4*sin(x)^3=combine(4*sin(x)^3,trig);
```

$$4\sin(x)^3 = -\sin(3x)+3\sin(x)$$

```
> exp(x)^2*exp(y)=combine(exp(x)^2*exp(y),exp);
```

$$(\mathbf{e}^{x})^{2}\,\mathbf{e}^{y} = \mathbf{e}^{(2x+y)}$$

```
> combine(Int(x,x=a..b)-Int(x^2,x=a..b));
```

$$\int_a^b x - x^2\,dx$$

```
> vyraz:=exp(sin(a)*cos(b))*exp(cos(a)*sin(b));
```

$$vyraz := \mathbf{e}^{(\sin(a)\cos(b))}\,\mathbf{e}^{(\cos(a)\sin(b))}$$

```
> combine(vyraz,[trig,exp]);
```

$$\mathbf{e}^{\sin(a+b)}$$

2.7 Převedení výrazů do jiného matematické tvaru - funkce `convert`

Funkce `convert` slouží pro převody výrazů do požadovaného tvaru. Některé konverze provedené touto funkcí jsou pouze převody mezi datovými typy. Jedná se například o převody mezi číselnými soustavami nebo ze stupňů na radiány. Jinou skupinu konverzí pomocí funkce `convert` tvoří převody algebraických výrazů.

Tato funkce se obvykle volá se dvěma parametry. První parametr obsahuje výraz, který má být upraven, druhý parametr specifikuje formu, na kterou má být výraz převeden.

■ Příklad 2.8:
Pro snazší pochopení funkce `convert` jsou v tomto příkladu uvedeny ukázky použití této funkce pro převod čísla do binárního tvaru, součet prvků v seznamu, převod výrazu obsahujícího desetinné číslo na výraz obsahující zlomek a rozklad zlomku na parciální zlomky.

```
> convert(9,binary);
```

$$1001$$

```
> convert([1,2,3,4],`+`);
```

$$10$$

```
> convert(1.23456*x,fraction);
```

$$\frac{3858}{3125}x$$

```
> (x^3+x)/(x^2-1)=convert((x^3+x)/(x^2-1),parfrac,x);
```

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

2.8 Stanovení podvýrazu z výrazu nebo rovnice - funkce `isolate`

Častým případem při manipulaci s matematickými výrazy bývá situace, kdy potřebujeme z rozsáhlého výrazu nebo rovnice vyjádřit některý v nich obsažený podvýraz (podvýrazem se samozřejmě rozumí i jednotlivé proměnné). V takovém případě je vhodné použít funkci `isolate`, která danou výraz nebo rovnici převede do tvaru, kde je na levé straně daný podvýraz a na pravé straně jeho hodnota vyjádřená pomocí ostatních proměnných z výrazu nebo rovnice.

Funkce `isolate` má tři parametry, z nichž první dva jsou povinné. První parametr obsahuje výraz respektive rovnici, z níž se bude podvýraz vyjadřovat. Pokud je v tomto parametru uveden výraz, předpokládá se implicitně, že má být roven nule. Jako druhý parametr musí být uveden podvýraz, který má být z rovnice vyjádřen. Třetím nepovinným parametrem může být přirozené číslo označující maximální počet úprav, které mohou být při výpočtu provedeny.

■ Příklad 2.9:
Zde je ukázáno použití funkce isolate.

```
>  isolate((x-b),x);
```

$$x=b$$

```
> isolate(4*x*sin(x)=3,sin(x));
```

$$\sin(x) = \frac{3}{4}\frac{1}{x}$$

```
> isolate(x^2-3*x-5,x^2);
```

$$x^2 = 3\,x + 5$$

Kapitola 3

Řešení rovnic

Stejně jako při „ručním“ řešení rovnic a nerovnic pomocí tužky a papíru existují i při jejich řešení s využitím systémů počítačové algebry dva základní přístupy. Buď můžeme hodnotu neznámé vypočítat nebo se můžeme uchýlit ke grafickému řešení rovnice. Ve druhém případě je výsledkem grafické znázornění hodnot řešení vyhovujících dané rovnici či soustavě rovnic. V praxi se nejčastěji tyto hodnoty řešení zobrazují v systému kartézských souřadnic, což plně postačí i pro potřebu tohoto kurzu. Následující text je věnován řešení rovnic pomocí systému počítačové algebry Maple, tak jak v předchozích kapitolách. Rovnice budeme řešit jak analyticky, tak numericky (v případě, že nelze analytické řešení vyjádřit) a rovněž pro větší názornost i graficky.

3.1 Analytické řešení rovnic

Podobně jako v ostatních oblastech matematiky, nabízí Maple 7 i v případě řešení rovnic několik příkazů, které můžeme použít. Ne všechny příkazy jsou však vhodné pro všechny typy rovnic.

3.1.1 Funkce `isolate`

Tomuto příkazu byla již věnována část předcházející kapitoly, kde se funkce `isolate` používala pro vyjadřování matematických výrazů obsažených v rovnicích. Omezíme-li se při obecném chápání funkce `isolate` pouze na vyjadřování jednotlivých proměnných, získáme tak nástroj pro řešení těchto rovnic.

■ Příklad 3.1:
Nejprve ukážeme řešení lineární rovnice, kde je řešením racionální číslo, které Maple 7 vyjádří přesně pomocí zlomku:

```
> isolate(3*x-7=0,x);
```

$$x = \frac{7}{3}$$

V případě lineárních rovnic se symbolickými koeficienty můžeme pomocí funkce `isolate` získat i jejich obecné řešení:

```
> isolate(a*x+b=0,x);
```

$$x = -\frac{b}{a}$$

Použití funkce `isolate` pro řešení kvadratických rovnic, případně dalších algebraických rovnic vyšších řádů, již není zcela úspěšné, neboť získáme jen jedno řešení. Funkce `isolate` určí pouze jeden kořen zadané rovnice a další kořen již nehledá. Stejně se tato funkce chová i v ostatních případech, kdy rovnice mají více kořenů.

```
> isolate(a*x^2+b*x+c=0,x);
```

$$x = \frac{1}{2}\,\frac{-b + \sqrt{b^2 - 4\,a\,c}}{a}$$

```
> isolate(x^2+3*x+2=0,x);
```

$$x = -2$$

■ Příklad 3.2:
Funkci `isolate` lze rovněž s úspěchem použít při řešení dalších typů rovnic, jako jsou například některé goniometrické nebo exponenciální rovnice.

```
> isolate(sin(2*x)=1,x);
```

$$x = \frac{1}{4}\pi$$

```
> isolate(exp(2*x)=4,x);
```

$$x = \frac{1}{2}\ln(4)$$

3.1.2 Funkce solve

Nejvhodnějším nástrojem pro řešení rovnic v systému Maple 7 je funkce `solve`. Hlavní výhodou funkce `solve` vzhledem k funkci `isolate` je možnost řešit soustavy rovnic o více neznámých. Funkce `solve` také narozdíl od `isolate` hledá i další kořeny u těch rovnic, které mají více řešení.

K funkci `solve` existuje několik jejích dalších variant (`msolve`, `isdolve`, `fsolve`, `dsolve`), které jsou použitelné v různých specifických případech. Pro potřeby tohoto kurzu však vystačíme se základní variantou `solve`. Podrobný popis zmiňovaných funkcí je uveden v nápovědě systému Maple 7.

Funkce `solve` se obvykle volá se dvěma parametry. Prvním parametrem je rovnice nebo soustava rovnic, která se má řešit, druhým je proměnná nebo proměnné, které chceme z rovnic vypočítat. Zadáme-li jako první parametr matematický výraz, implicitně se tento výraz položí roven nule a takto vzniklá rovnice se dále řeší. Pokud nezadáme druhý parametr, rovnice se automaticky vyřeší pro všechny proměnné, které obsahuje. V případě, kdy rovnice obsahuje více než jednu proměnnou, získáme někdy pro některou z proměnných nic neříkající výsledek typu $x=x$.

■ Příklad 3.3:

Na příkladu hledání řešení pro proměnnou b jsou ukázány rozdílné výsledky, které získáme bez použití, případně s použitím druhého parametru funkce `solve`. Při řešení rovnice s číselnými koeficienty Maple 7 dává přesně řešení, kde se nám vyskytuje ve zlomku i odmocnina. Chceme-li získat přibližné řešení použijeme k vyhodnocení řešení funkce `evalf`.

```
>  solve(a+b=c);
```

$$\{c = c, b = b, a = -b + c\}$$

```
>  solve(a+b=c,b);
```

$$-a + c$$

```
> solve(6*x-3=sqrt(2)*x);
```

$$-3\,\frac{1}{-6+\sqrt{2}}$$

```
> evalf(%);
```

$$.6541953144$$

Poznámka: V případě řešení soustavy rovnic zadává se jako první parametr funkce `solve` množina rovnic nebo výrazů a jako druhý parametr množina proměnných. Množina v Maple 7 s zapisuje jako posloupnost jejich prvků do složených závorek. Druhý parametr funkce `solve` můžeme případně opět vynechat. Výsledek funkce je pak vrácen jako množina rovnic, kde její prvky mají tvar *proměnná = výraz.*

■ Příklad 3.4:
Vyřešme soustavu tří lineárních rovnic:

```
> soustava_rovnic:={u+v+w=1, 3*u+v=3, u-2*v-w=0};
```

$$soustava\_rovnic := \{u + v + w = 1, 3u + v = 3, u - 2v - w = 0\}$$

```
> solve(soustava_rovnic);
```

$$\{w = \frac{-2}{5}, u = \frac{4}{5}, v = \frac{3}{5}\}$$

Jak již bylo naznačeno výše, můžeme příkaz `solve` úspěšně použít také pro obecné vyjádření kořenů algebraických rovnic.

■ Příklad 3.5:
Ukážeme na tomto příkladu, že na rozdíl od funkce `isolate` nalezne funkce `solve` všechny kořeny obecné kvadratické a kubické rovnice.

```
>  solve(a*x^2+b*x+c,x);
```

$$\frac{1}{2}\frac{-b+\sqrt{b^2-4ac}}{a}, \frac{1}{2}\frac{-b-\sqrt{b^2-4ac}}{a}$$

```
>  solve(x^3+p*x^2+q*x+r,{x});
```

$$\left\{x = \frac{1}{6}(36qp - 108r - 8p^3 + 12\sqrt{12q^3 - 3q^2p^2 - 54qpr + 81r^2 + 12rp^3})^{(1/3)} - \frac{6\left(\frac{1}{3}q - \frac{1}{9}p^2\right)}{(36qp - 108r - 8p^3 + 12\sqrt{12q^3 - 3q^2p^2 - 54qpr + 81r^2 + 12rp^3})^{(1/3)}} - \frac{1}{3}p\right\},$$

$$\left\{x = -\frac{1}{12}(36qp - 108r - 8p^3 + 12\sqrt{12q^3 - 3q^2p^2 - 54qpr + 81r^2 + 12rp^3})^{(1/3)} + \frac{3\left(\frac{1}{3}q - \frac{1}{9}p^2\right)}{(36qp - 108r - 8p^3 + 12\sqrt{12q^3 - 3q^2p^2 - 54qpr + 81r^2 + 12rp^3})^{(1/3)}} - \frac{1}{3}p + \frac{1}{2}I\sqrt{3}\left(\frac{1}{6}(36qp - 108r - 8p^3 + 12\sqrt{12q^3 - 3q^2p^2 - 54qpr + 81r^2 + 12rp^3})^{(1/3)}\right.\right.$$

$$+\frac{6\left(\frac{1}{3}q-\frac{1}{9}p^2\right)}{(36\,q\,p-108\,r-8\,p^3+12\sqrt{12\,q^3-3\,q^2\,p^2-54\,q\,p\,r+81\,r^2+12\,r\,p^3})^{(1/3)}}\Bigg\},\Bigg\{$$

$$x=-\frac{1}{12}(36\,q\,p-108\,r-8\,p^3+12\sqrt{12\,q^3-3\,q^2\,p^2-54\,q\,p\,r+81\,r^2+12\,r\,p^3})^{(1/3)}$$

$$+\frac{3\left(\frac{1}{3}q-\frac{1}{9}p^2\right)}{(36\,q\,p-108\,r-8\,p^3+12\sqrt{12\,q^3-3\,q^2\,p^2-54\,q\,p\,r+81\,r^2+12\,r\,p^3})^{(1/3)}}-\frac{1}{3}p$$

$$-\frac{1}{2}I\sqrt{3}\Bigg($$

$$\frac{1}{6}(36\,q\,p-108\,r-8\,p^3+12\sqrt{12\,q^3-3\,q^2\,p^2-54\,q\,p\,r+81\,r^2+12\,r\,p^3})^{(1/3)}$$

$$+\frac{6\left(\frac{1}{3}q-\frac{1}{9}p^2\right)}{(36\,q\,p-108\,r-8\,p^3+12\sqrt{12\,q^3-3\,q^2\,p^2-54\,q\,p\,r+81\,r^2+12\,r\,p^3})^{(1/3)}}\Bigg)\Bigg\}$$

■ Příklad 3.6:

Na příkladu rovnice 4. řádu ukážeme jak lze ukládat její řešení do seznamu `reseni` a pak s jednotlivá řešení používat např. když zkontrolujeme správnost třetího řešení jeho dosazením do původní rovnice. K tomu jsme využili mapleovské funkce `subs`, která do svého druhého parametru - výrazu `rovnice` - dosadí za proměnnou `x` třetí řešení rovnice 4. řádu uložené v prvku seznamu `reseni[3]`.

```
> rovnice := x^4-5*x^2+6*x=2;
```

$$rovnice := x^4-5\,x^2+6\,x=2$$

```
> reseni:=[solve(rovnice,x)];
```

$$reseni := [1,\ 1,\ -1+\sqrt{3},\ -1-\sqrt{3}\,]$$

```
> evalf(reseni);
```

$$[1.,\ 1.,\ .732050808,\ -2.732050808\,]$$

```
> subs( x=reseni[3], rovnice );
```

$$(-1+\sqrt{3}\,)^4-5\,(-1+\sqrt{3}\,)^2-6+6\sqrt{3}=2$$

```
> simplify(%);
```

$$2=2$$

■ Příklad 3.7:

Pomocí funkce `solve` lze řešit i trigonometrické rovnice:

```
> solve(cos(x)^2=1, x);
```

$$\pi,\ 0$$

```
> solve( sin(x)=cos(x)-1, x );
```

$$-\frac{1}{2}\pi,\ 0$$

```
> solve( cos(x)+y = 9, x );
```

$$\pi - \arccos(y - 9)$$

■ **Příklad 3.8:**

V tomto příkladu ukážeme způsob použití funkce `solve` při řešení soustav lineárních a kvadratických nerovnic:

```
> solve({ x+y > 0, x-y <= 1, y = 2 }, {x,y});
```

$$\{y = 2,\ -2 < x,\ x \le 3\}$$

```
> solve( {x^2-x-2<=0,x^2-x>0}, x );
```

$$\{x < 0,\ -1 \le x\},\ \{1 < x,\ x \le 2\}$$

Příkazy typu solve hledají řešení rovnic v celém komplexním oboru. Chceme-li se omezit na řešení v oboru reálném, použijeme k tomu jednu z novinek Maplu 7 - balík RealDomain - jedním z následujících tří způsobů:

```
> solve(x^3+x=0,x);
```

$$0,\ I,\ -I$$

```
> use RealDomain in solve(x^3+x=0,x);
> end;
```

$$0$$

```
> ln(-1);
```

$$I\,\pi$$

```
> RealDomain[ln](-1);
```

undefined

```
> with(RealDomain);
Warning, these protected names have been redefined and unprotected: Im, Re,
^, arccos, arccosh, arccot, arccoth, arccsc, arccsch, arcsec, arcsech,
arcsin, arcsinh, arctan, arctanh, cos, cosh, cot, coth, csc, csch, eval,
exp, expand, limit, ln, log, sec, sech, signum, simplify, sin, sinh, solve,
sqrt, surd, tan, tanh
```

[$\Im$, $\Re$, ^, arccos, arccosh, arccot, arccoth, arccsc, arccsch, arcsec, arcsech, arcsin, arcsinh, arctan, arctanh, cos, cosh, cot, coth, csc, csch, *eval*, exp, *expand*, *limit*, ln, log, sec, sech, signum, *simplify*, sin, sinh, *solve*, sqrt, *surd*, tan, tanh]

```
> ln(-1);
```

undefined

3.2 Grafické řešení rovnic a nerovností

Jak již bylo uvedeno v části věnované vizualizaci matematických objektů, má Maple rozsáhlé možnosti pro zobrazování grafů funkcí. Toho lze úspěšně využít při grafickém řešení rovnic a nerovnic. Zde nám Maple 7 nabízí možnost, jak rychle a přesně zobrazit daný objekt, což oceníme především v případě nelineárních rovnic, jejichž „ruční“ grafické řešení může být

zdlouhavé a mnohdy vede k nepřesnostem. Pro zobrazování funkcí, jejichž rovnice budeme řešit použijeme příkazy standardní knihovny `plots`.

Velkou výhodou tohoto přístupu je také možnost zobrazovat grafy i bez předchozího hlubšího studia ostatních funkcí systému Maple 7. Většina jeho zobrazovacích možností je rychle a snadno dostupná. Nejjednodušším způsobem jak vytvořit graf, je kliknout pravým tlačítkem myši na vybranou funkci a z kontextového menu Maple 7 vybrat položku `plots`. Program již sám automaticky nastaví ostatní parametry grafu (souřadnice, barvy, rozsahy …), které však lze kdykoliv změnit podle potřeby.

3.2.1 Grafické řešení rovnic

Grafické řešení rovnic spočívá v tom, že do jednoho grafu umístíme jak křivku dané funkce, tak i body, které vyhovují řešení její rovnice. V případě rovnic o jedné neznámé, jejichž řešení zobrazujeme v dvourozměrných kartézských souřadnicích, umísťujeme kořeny rovnice zpravidla na osu *x*. Má-li například nějaká rovnice kořen 1, bude tento kořen znázorněn jako bod o souřadnicích [1,0]. Na následujícím řešeném příkladě je ukázán postup při vizualizaci řešení kvadratické rovnice.

■ Příklad 3.9:

Pokud chceme v Maple 7 vykreslovat grafy, je vhodné nejdříve zavést grafickou knihovnu plots pomocí následujícího příkazu:

```
> with(plots):
```

```
Warning, the name changecoords has been redefined
```

```
> f:=x^2+2*x-3;
```

$$f := x^2 + 2x - 3$$

```
> k := [solve(f)];
```

$$k := [1, -3]$$

Předchozí dva příkazy uložily do proměnné `f` rovnici (výraz) a nalezly její kořeny. Kořeny rovnice se uložily jako seznam do proměnné `k`. Jak je vidět, rovnice má dva kořeny o hodnotách 1 a -3. Do proměnné `pf` uložíme graf dané funkce.

Do proměnné *pk* se uloží grafické znázornění kořenů rovnice.

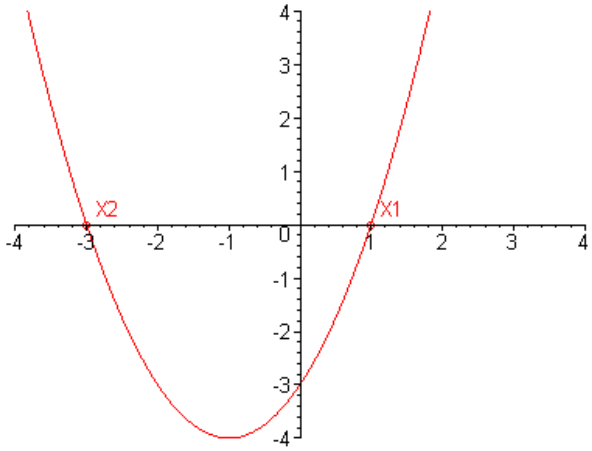
```
> pf:=plot(f,x=-4..4,y=-4..4):
> pk:=plot([[k[1],0],[k[2],0]],style=point,symbol=circle):
```

Do proměnných `pt1` a `pt2` uložíme popisky kořenů rovnice, které se mají zobrazit. V tomto případě jsou kořeny označeny jako X1 a X2. Popisky kořenů jsou zde posunuty ve směru obou os o hodnotu 0,3. To je z důvodu, aby se popisky nepřekrývaly s křivkou a osami souřadnic.

```
> pt1:=textplot([k[1]+0.3,0.3,'X1']):
> pt2:=textplot([k[2]+0.3,0.3,'X2']):
```

Na závěr se vše zobrazí do jednoho grafu pomocí příkazu `display`.

```
> display({pf,pk,pt1,pt2});
```



■ **Příklad 3.10:**
Na tomto příkladu ukážeme postup při řešení rovnice *tan (x + 1) – 1 = 0* na intervalu <-1,1>.

Nejprve připomeňme, že funkce *tan* není na celé reálné ose spojitá a má body nespojitosti (dokonce 2. druhu). Pro její korektní zobrazení je nutné u mapleovské funkce `plot` nastavit její parametr `discont` na hodnotu `true`.

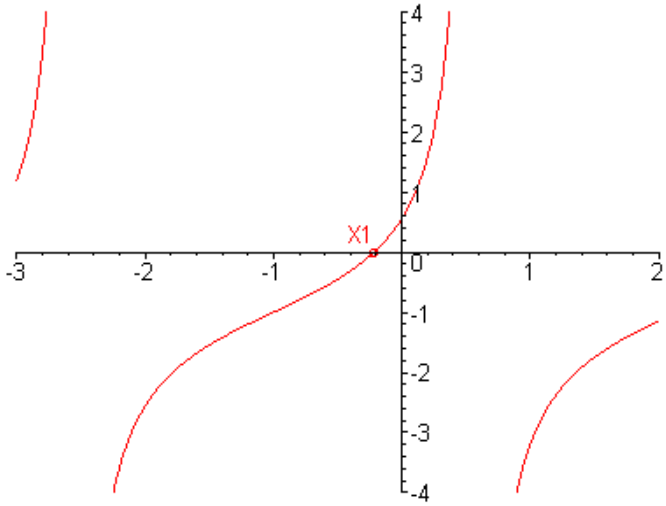
```
> with(plots):
> f:=tan(x+1)-1;
```

$$f := \tan(x+1) - 1$$

```
> k:=[solve(f)];
```

$$k := \left[\frac{1}{4}\pi - 1\right]$$

```
> pf:=plot(f,x=-3..2,y=-4..4, discont=true):
> pk:=plot([[k[1],0]],style=point,symbol=circle):
> pt:=textplot([k[1]-0.1,0.3,'X1']):
> display({pf,pk,pt});
```



3.2.1 Grafické řešení nerovností – funkce `inequal`

V Maple 7 je umožněno graficky zobrazit řešení soustavy lineárních nerovností dvou proměnných v rovině pomocí funkce `inequal`. Počítačový graf řešení této soustavy sestává ze čtyř částí, kde u každá části je možno si zvolit její barvu pomocí speciálního parametru. Tyto parametry jsou:

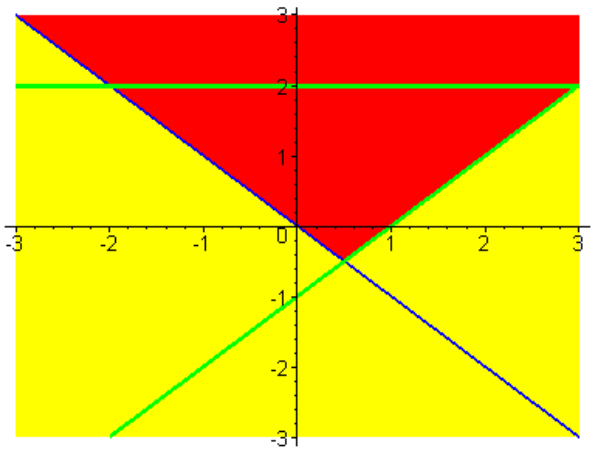
- `optionsfeasible` – pro oblast řešení, která splňuje všechny nerovnosti,
- `optionsexcluded` – pro oblasti, kde není splněna alespoň jedna nerovnost,
- `optionsclosed` - pro přímky, reprezentující hranice ostrých nerovností,
- `optionsopen` - pro přímky, reprezentující hranice neostrých nerovností a průsečíky těchto přímek.

Styly zobrazení těchto čtyř oblastí se mohou volit nezávisle na sobě. Ukážeme to v následujícím příkladu.

■ Příklad 3.11:

Nejprve nalezneme grafické řešení soustavy nerovností z příkladu 3.8

```
> with(plots): inequal( { x+y > 0, x-y <= 1, y = 2 }, x=-3..3,
y=-3..3,optionsfeasible=(color=red), optionsopen=(color=blue,
thickness=2), optionsclosed=(color=green, thickness=3),
optionsexcluded=(color=yellow) );
```

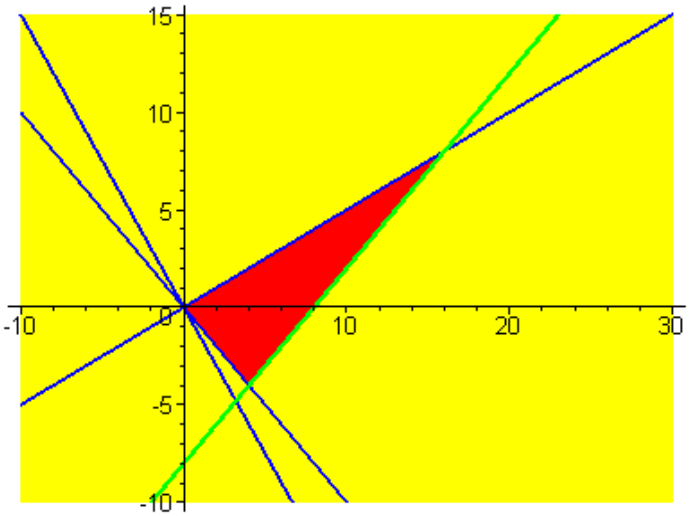


Jako další příklad vyřešíme následující soustavu nerovností

```
> soustava_nerovnosti:={a+b>3, 2*b-a<6, 3*a+2*b>5, -b+a<=8,
3*a+2*b>0};
```

$$soustava\_nerovnosti := \{3 < a + b,\ 5 < 3\,a + 2\,b,\ -b + a \le 8,\ 0 < 3\,a + 2\,b,\ 2\,b - a < 6\}$$

```
> inequal( soustava_nerovnosti,a=-10..30, b=-10..15,
optionsfeasible=(color=red), optionsopen=(color=blue,
thickness=2), optionsclosed=(color=green, thickness=3),
optionsexcluded=(color=yellow));
```


15
10
5
0
-5
-10
-10
10
20
30

Kapitola 4

Úvod do matematické analýzy

V oblasti matematické analýzy nabízí Maple pestrou škálu příkazů. My se z této nabídky omezíme pouze na tři základní části matematické analýzy. Jsou to:

- **Limity funkcí**
- **Derivace funkcí**
- **Integrály funkcí**

Dále se ve všech třech oblastech budeme věnovat pouze funkcím jedné proměnné.

4.1 Limity funkcí jedné proměnné

Pro výpočet limity funkce slouží v systému Maple příkaz `limit`. V základní podobě má příkaz `limit` dva parametry. První parametr udává funkci, pro kterou je limita počítána. Druhý parametr definuje hodnotu proměnné, pro níž se limita počítá. Je-li limita počítána pro hodnotu proměnné jdoucí ke kladnému respektive zápornému nekonečnu, definuje se tato hodnota jako `+infinity` respektive `-infinity`.

■ **Příklad 4.1:**
V tomto příkladu se ukazuje užití příkazu `limit` se dvěma základními parametry.

```
> limit(sin(x)/x, x=0);
```

$$1$$

```
> limit(exp(x), x=infinity);
```

$$\infty$$

```
> limit(exp(x), x=-infinity);
```

$$0$$

```
> limit(1/x, x=0);
```

undefined

Jak je vidět, v případě lomené funkce z předcházejícího příkladu limita v bodě *x=0* neexistuje. Pro tutu funkci však existují v bodě *x=0* limita zprava i zleva. To, že má Maple počítat limitu zprava nebo zleva, mu sdělíme pomocí třetího nepovinného parametru. Pomocí tohoto parametru můžeme programu také sdělit číselný obor (reálná nebo komplexní čísla),ve kterém se má limita počítat.

■ **Příklad 4.2:**
Použití třetího parametru si opět ukážeme na lomené funkci.

```
> limit(1/x, x=0, left);
```

$$-\infty$$

```
> limit(1/x, x=0, right);
```

$$\infty$$

```
> limit(1/x, x=0, real);
```

undefined

```
> limit(1/x, x=0, complex);
```

$$\infty - \infty\, I$$

Pro zefektivnění výstupů můžeme kombinovat příkaz `limit` s příkazem `Limit`, který danou limitu vypíše v „tradiční“ matematické formě. Parametry příkazu `Limit` jsou totožné s parametry `limit`.

■ Příklad 4.3:
Příklad ukazuje použití příkazu `Limit`.

```
> Limit(2/(3*x), x=infinity)=limit(2/(3*x), x=infinity);
```

$$\lim_{x \to \infty} \frac{2}{3}\frac{1}{x} = 0$$

S funkcí `Limit` lze pracovat jako s matematickým výrazem. Je možné ji například upravovat pomocí příkazu `combine`.

```
> combine(Limit(1/x,x=0)*Limit(x,x=0));
```

$$\lim_{x \to 0} 1$$

4.2 Derivace funkcí jedné proměnné

Chceme-li v Maple derivovat nějakou funkci, použijeme k tomu příkaz `diff`. Jako první parametr se u tohoto příkazu uvádí funkce, kterou chceme derivovat. Jako další parametry se uvádí proměnné, podle nichž se bude derivovat. V případě derivací funkce jedné proměnné to znamená, že kolikrát danou proměnnou uvedeme, tolikátou derivaci bude Maple počítat. Chceme-li například spočítat třetí derivaci funkce *f*, bude mít příkaz `diff` tvar `diff(`*f*`,x,x,x)`. Případně můžeme použít zkrácenou formu zápisu `diif(`*f*`,x$3)`.

■ Příklad 4.4:

Příklad ukazuje výpočet 1., 2. a 3. derivace funkce *sin(x)*.

```
> diff(sin(x),x);
```

$$\cos(x)$$

```
> diff(sin(x),x,x);
```

$$-\sin(x)$$

```
> diff(sin(x),x$3);
```

$$-\cos(x)$$

Stejně jako k příkazu `limit` existuje příkaz `Limit`, existuje i k `diff` příkaz `Diff`. Parametry obou příkazů jsou opět shodné.

■ Příklad 4.5:

V tomoto příkladě ukážeme jak je vhodné používat příkaz `Diff`:

```
> Diff(x^7+3*x^3-sin(x),x$2)=diff(x^7+3*x^3-sin(x),x$2);
```

$$\frac{\partial^2}{\partial x^2}\left(x^7 + 3\,x^3 - \sin(x)\right) = 42\,x^5 + 18\,x + \sin(x)$$

4.3 Integrály funkcí jedné proměnné

Integrování v systému Maple se provádí pomocí funkce `int`. Tato funkce se použije jak v případě, kdy chceme spočítat neurčitý integrál, tak i v případě, kdy se nám jedná o integrál určitý. V prvním případě je výsledkem příkazu funkce, ve druhém příkaz vrací konkrétní hodnotu nebo výraz (obsahuje-li integrovaná funkce parametry).

Nejdříve si ukážeme způsob, jakým se pomocí příkazu `int` počítá neurčitý integrál. V takovém případě se příkaz `int` volá se dvěma parametry. Prvním parametrem je integrovaná funkce, druhým je proměnná, podle které se integruje.

■ **Příklad 4.6:**
Výpočet neurčitého integrálu:

```
> int(-x^2+3,x);
```

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

```
> int(2+exp(x),x);
```

$$2x+\mathbf{e}^x$$

Pokud chceme spočítat pomocí příkazu `int` určitý integrál, přiřadíme k proměnné ve druhém parametru integrační interval. Význam prvního parametru zůstává stejný jako v případě neurčitého integrálu.

■ **Příklad 4.7:**
Výpočet určitého integrálu:

```
> int(-x^2+3,x=-1..1);
```

$$\frac{16}{3}$$

```
> int(2+exp(x),x=1..2);
```

$$2+\mathbf{e}^2-\mathbf{e}$$

Také k příkazu `int` existuje příkaz `Int` a má stejný význam jako u ostatních příkazů.

■ **Příklad 4.8:**
Použití příkazu `Int`:

```
> Int(sin(x),x)=int(sin(x),x);
```

$$\int \sin(x)\,dx=-\cos(x)$$

```
> Int(-x^2+3,x=-1..1)=int(-x^2+3,x=-1..1);
```

$$\int_{-1}^{1} -x^2+3\,dx=\frac{16}{3}$$

```
> Int(2+exp(x),x=1..2)=int(2+exp(x),x=1..2);
```

$$\int_{1}^{2} 2+\mathbf{e}^x\,dx=2+\mathbf{e}^2-\mathbf{e}$$

Literatura

[1] AUER V. - Vědecké výpočty na střední škole, diplomová práce, Masarykova univerzita v Brně, Brno 2000
[2] BUCHAR J., HŘEBÍČEK J., HŘEBÍČKOVÁ J., SLABĚNÁKOVÁ J. - Úvod do programového souboru MAPLE V. Skripta Vysoká škola zemědělská v Brně, Brno 1994
[3] DOČKAL J.- Výuka matematiky s podporou systému MAPLE V, příspěvek ve Sborníku 24. Konference o matematice na VŠTEZ Brno, Brno 1996
[4] DOŠLÁ Z., PLCH R., SOJKA P. - Matematická analýza s programem Maple - Diferenciální počet funkcí více proměnných, Masarykova univerzita v Brně, Brno 1999
[5] GANDER W., HŘEBÍČEK J.- Solving Problems in Scientific Computing Using Maple and MATLAB. Springer-Verlag, Berlin 1997
[6] HEAL K. M., HANSEN M.L., RICKARD K.M. - Maple 6 Learning Guide, Waterloo Maple Inc., Waterloo 2000
[7] JOBÁKOVÁ D. - Praktikum z počítačů ve výuce matematiky, diplomová práce, Masarykova univerzita v Brně, Brno 1997
[8] PLCH R. – Diferenciální počet funkcí více proměnných s programem MapleV, disertační práce, Masarykova univerzita v Brně, Brno 1998
[9] SVÍTIL F. - Sbírka příkladů pro využití Maple V při výuce matematiky na středních školách, bakalářská práce, Masarykova univerzita v Brně, Brno 2001
[10] http://www.maplesoft.com/ - stránky Waterloo Maple Inc.