Univerzita Karlova v Praze

Pedagogická fakulta

BAKALÁŘSKÁ PRÁCE

Klára Krejčíčková

Možnosti využití programu Wolfram Mathematica ve výuce matematiky

Katedra matematiky a didaktiky matematiky

Vedoucí bakalářské práce: RNDr. Antonín Jančařík, Ph.D.

Studijní program: Specializace v pedagogice

Studijní obor: Jednooborové studium – Matematika

Praha 2014

Děkuji svému vedoucímu RNDr. Antonínu Jančaříkovi, Ph.D. za rady, nápady a připomínky při korekturách bakalářské práce. Děkuji také své mámě za podporu a korektury při psaní mé bakalářské práce. Také bych chtěla poděkovat celé rodině za podporu během studia a tvorbu potřebného zázemí. Můj dík patří rovněž přátelům za podporu při psaní bakalářské práce.

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V Praze dne 10. 4. 2014 Klára Krejčíčková

Název práce: Možnosti využití programu Wolfram Mathematica ve výuce matematiky

Autor: Klára Krejčíčková

Vedoucí bakalářské práce: RNDr. Antonín Jančařík, Ph.D.

Abstrakt: Cílem práce je analyzovat možnosti použití programu Wolfram Mathematica ve výuce matematiky. Tyto možnosti jsou podpořeny vytvořenými applety v tomto programu. Čtenář se seznámí s prostředím tohoto programu a webovou podporou (včetně návodů a již hotových materiálů). Důraz je kladen na tvorbu demonstrací v programu Wolfram Mathematica. Část práce také seznamuje čtenáře s programy Wolfram Alpha, Wolfram Demonstration Project a Wolfram Problem Generator.

Klíčová slova: Výuka matematiky, CAS, Mathematica

Title: Possibilities of using Wolfram Mathematica program in Mathematics Teaching

Author: Klára Krejčíčková

Supervisor: RNDr. Antonín Jančařík, Ph.D.

Abstract: The aim of the thesis is analyzing of possibilities in application of program Wolfram Mathematica in education of Math. All analysed options are supported by applets created in investigated program. The text is focused on introduction of background of the program and web support including instructions and all prepared materials. The thesis mainly emphasizes on creation of demonstrations in program Wolfram Mathematica. Particular parts of the text are also focused on programs such as Wolfram Alpha, Wolfram Demonstration Project a Wolfram Problem Generator.

Keywords: mathematics Education, CAS, Mathematica

Obsah

Úvod

Trendy výuky se mění, poslední roky je tomu tak, že se školy snaží maximálně využívat výpočetní techniku – počítače, projektory, interaktivní tabule.

Cílem mé práce je, aby následující kapitoly pomohly učitelům a následně žákům a studentům oblíbit si matematiku případně i program Wolfram Mathematica a tím alespoň částečně změnit současný trend pouze na výběr humanitních vysokých škol, kde se matematikou nemusejí zabývat.

Program Wolfram Mathematica umožňuje tvorbu vlastních výukových materiálů (případně i sbírek), které může učitel matematiky při výuce matematiky používat. Program Wolfram Mathematica je placená, ale školní licence lze pořídit za příznivých finančních podmínek.

Tento software mě již před pár lety ohromil rozsahem svých možností. Dá se využívat jako kalkulačka, nebo pomocí něj můžeme vytvářet jednoduché aplikace či programovat složitější programy a řešit tak zajímavé problémy. Díky zkušenostem, nejen z učitelské praxe, ale i s tímto softwarem, jsem se rozhodla využít ve výuce program Wolfram Mathematica a další užitečné aplikace firmy Wolfram do výuky matematiky, tedy nejen do vyučovacích hodin, ale i jako mnohem zajímavější a pro dnešní děti i smysluplnější domácí přípravu na hodiny matematiky.

Ve své bakalářské práci se zaměřím hlavně na program Wolfram Mathematica, pokusím se vysvětlit, co nabízí a jak jej můžeme používat. Pokusím se přiblížit funkce v českém jazyce, aby i anglicky nemluvící kantoři měli možnost se programu Wolfram Mathematica začít věnovat.

Seznámím se i s online aplikacemi Wolfram Alpha, Wolfram Demonstration, Wolfram Problem Generator, popíšu jejich možnosti a pokusím se najít pro ně využití ve výuce, nastínit možnosti použití, ukázat příklady, které mohou inspirovat pro další využití.

Tématice programu Wolfram Mathematica ve výuce nejen matematiky se již několik středoškolských i vysokoškolských učitelů věnuje. Z jejich webových stránek jsem také čerpala náměty a nápady a zároveň jsem se snažila vymyslet v dané problematice něco nové, co by mohlo čtenářům pomoci osvojit práci s tímto programem. Rozhodla jsem se tedy věnovat především demonstracím, které jsou

zveřejnitelné na internetu na stránkách Wolfram Demonstration a tedy přístupné všem. Vyberu si středoškolská témata či témata vhodná pro základní školy, která se pokusím pomocí demonstrací přiblížit. Některé demonstrace budou, hlavně pro usnadnění práce učitelů, generátory úloh. Tyto bude moci používat každý učitel, který o to bude mít zájem. Pokusím se přiblížit práci s programem Wolfram Mathematica a proces tvorby těchto demonstrací.

Cílem mé práce je podat základní návod na tvorbu demonstrací, vytvořit demonstrace či jiné programy, které by následně mohly najít uplatnění ve výuce matematiky na základních či středních školách.

Práce je rozdělena do tří základních kapitol, každá má své podkapitoly. První kapitola se věnuje základnímu představení produktů firmy Wolfram, jako je Wolfram Mathematica, Wolfram Alpha, Wolfram Problem Generator, Wolfram Demonstration Project. Tyto produkty představím, zároveň ke každému projektu nabízím několik nápadů, jak je využít ve výuce matematiky.

V druhé kapitole se zaměřím na samotný program Wolfram Mathematica. Naučím se v ní základní operace a použití základních funkcí. Ukáži, že lze využít i znalostí z programování a tedy programovat. A hlavně vysvětlím, jak vytvářet ony demonstrace, jak je ukládat na internet, aby byly přístupné široké veřejnosti. Na mých demonstracích a programech přiblížím použití některých funkcí a příkazů. Pro doplnění představím jeden složitější program, který má sloužit jako námět pro případné další samostudium.

V třetí kapitole můžeme najít další zdroje inspirací pro tvorbu vlastních prací v programu Wolfram Mathematica. Upozorním na některé internetové stránky, které obsahují zajímavé a již hotové práce, které jsou taky přístupné všem. Podíváme se na to, co přináší nová verze programu Wolframu Mathematica, nyní již devátá, a co za novinky nabízí. A shrneme si, jak tedy produkty firmy Wolfram Research můžeme použít ve výuce matematiky.

Vše co v následujících kapitolách přiblížím, také sama používám. Nejen jako přípravu pro vyučovací hodiny, ale také na tábory a soustředění zaměřená na matematiku, která jsem dříve zažívala jako dítě a později jako vedoucí a program Wolfram Mathematica mi několikrát pomohl.

1. Produkty firmy Wolfram

1.1 Co je Wolfram Mathematica

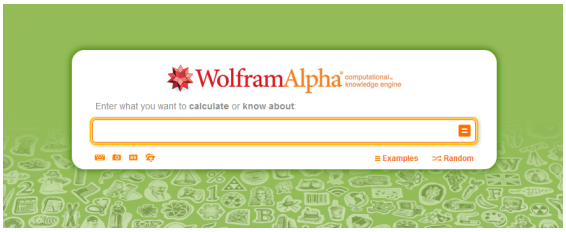
Program Wolfram Mathematica je jedním z nejrozšířenějších softwarových systémů pro provádění výpočtů a vizualizaci dat. Manipulace s grafickými objekty je snadná a proto se hojně používá při výuce nejen matematiky, ale i fyziky, chemie a dalších přírodovědných oborů. Použití programu Wolfram Mathematica ve vyučování přináší výraznou podporu při výkladu učiva nejen proto, že je kompatibilní s interaktivní tabulí. Žáci mohou doma používat výukové materiály, které obsahují dynamické prvky pomáhající porozumění učivu.

Program Wolfram Mathematica je kompletně v angličtině, což u žáků rozvíjí znalost tohoto jazyka a v dnešní době to není překážkou. Má velmi dobře zpracovanou nápovědu, která je dynamická a umožňuje si v ní rovnou cokoli vyzkoušet. Program Wolfram Mathematica se dá využít jako kalkulačka. Umí pracovat se symbolickými i numerickými výpočty, počítat rovnice, graficky zobrazovat zadané závislosti.

Pod firmou Wolfram vznikly další produkty – Wolfram Alpha a Wolfram Demonstration. Úplnou novinkou je Wolfram Problem Generator. Tyto software či aplikace se pokusím přiblížit a ukázat jejich možnosti použití ve výuce matematiky, jak v hodinách, tak v domácí přípravě.

1.2 Wolfram Alpha

Wolfram Alpha je internetová aplikace. Na internetové stránce najdeme řádek (Obr. 1.1), kam zadáváme požadavek v podobném formátu jako do programu Wolfram Mathematica a Wolfram Alpha jej zpracuje. Výhodou této webové aplikace je, že není potřeba nic instalovat, protože



Obrázek 1.1: Wolfram Alpha

funguje přímo z internetového prohlížeče. Wolfram Alpha nabízí možnost nainstalovat si do prohlížeče rozšíření, které nám umožňuje zadávat dotaz pro Wolfram Alpha rovnou do vyhledávače internetových adres, ušetříme si tak zadávání jedné internetové adresy. Wolfram Alpha již za nízký poplatek existuje jako aplikace do chytrých mobilních telefonů a tabletů.

Wolfram Alpha můžeme používat ve třech módech: bez přihlášení, s přihlášením, s přihlášením a následným uhrazením nízkého poplatku. Vytvoření účtu je bezplatné. Následné přihlášení nám otevírá nové možnosti: řešení krok za krokem (třikrát za den), analýza osobních dat z Facebookového účtu, historie a oblíbené funkce. Pokud máme vlastní účet, nabízí se možnost vyzkoušet zdarma na sedm dní Wolfram Alpha Pro. Další použití této aplikace je zpoplatněno spíše symbolickou částkou. Wolfram Alpha Pro umožňuje neomezené zobrazování řešení krok za krokem, vkládání dat, souborů, obrázků, stahování dat ve více než 60 podporovaných formátech, přizpůsobování grafů svým potřebám a další.

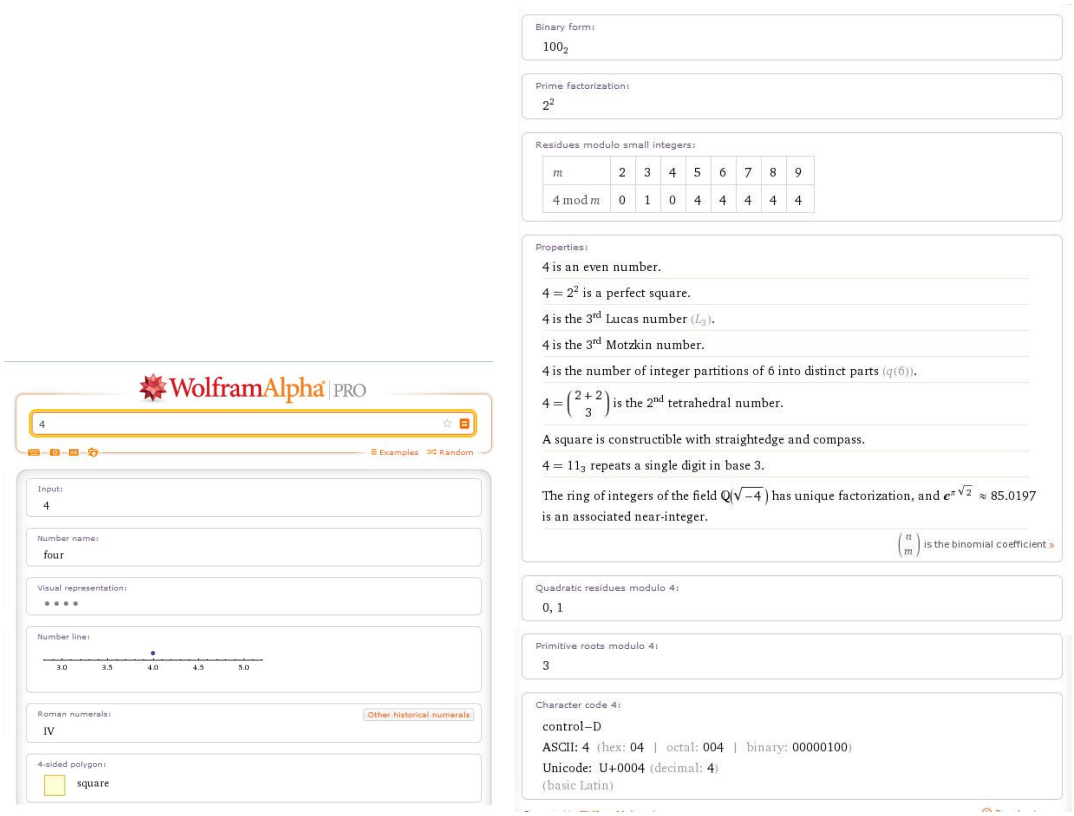
Wolfram Alpha je vhodná i pro lidi, kteří se matematice nevěnují. Je to vlastně taková zajímavá interaktivní encyklopedie. Po zadání slova, jména významného člověka, data, nám podá podrobné a zajímavé informace z celého oboru, které klíčové slovo zahrnuje. Wolfram Alpha je tedy všestranný pomocník pro rozvíjení znalostí nejen z matematiky. Pokusím se přiblížit hlavně tu matematickou stránku této aplikace.

1.2.1 Wolfram Alpha a matematika

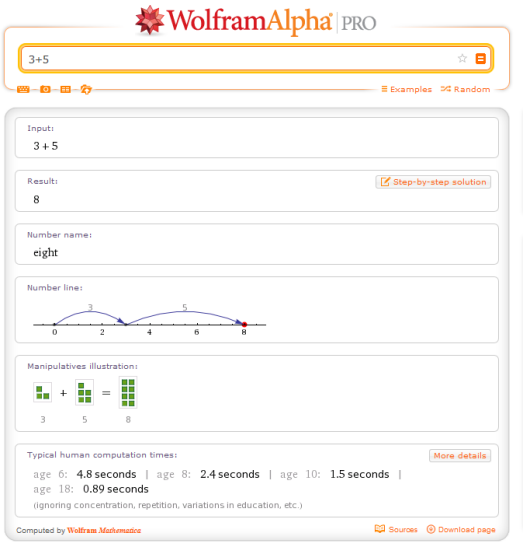
Podíváme se na to, co Wolfram Alpha nabízí. Zaměříme se na matematické dovednosti, ale ukážeme si, že možnosti použití jsou větší.

Nejjednodušší je vložit číslo, po zadání tohoto dotazu nabízí Wolfram Alpha spoustu interpretací. Napíše anglický název čísla, zobrazí ho na číselné ose, napíše ho pomocí římských číslic, pomocí dvojkové soustavy, provede faktorizaci a vypíše další zajímavé vlastnosti zadaného čísla jako můžeme vidět na obrázku (Obr. 1.2).

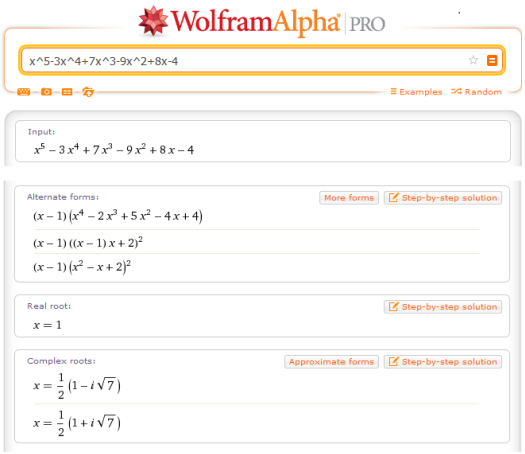
Po zadání základních matematických operací $+, -, \cdot, :$ vypíše výsledek, ilustruje příklad na obrázcích, číselné ose (Obr. 1.3).



Obrázek 1.2: WA po zadání čísla 4



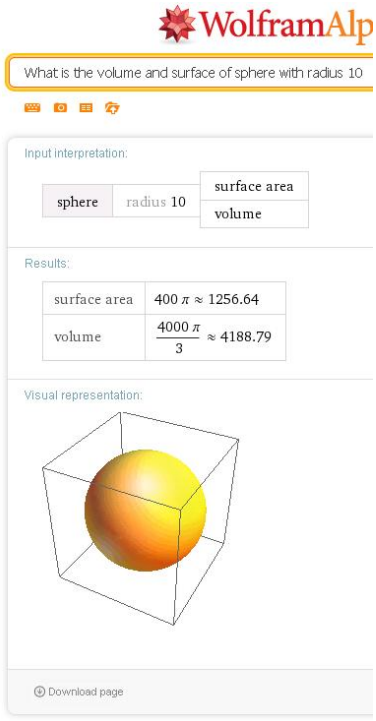
Obrázek 1.3: Součet dvou čísel



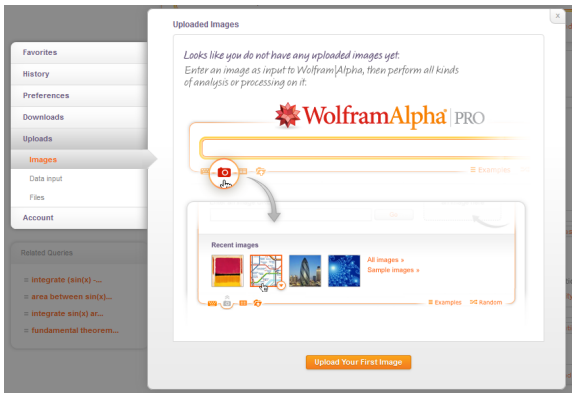
Obrázek 1.4: Počítání s výrazy

Wolfram Alpha počítá integrály, derivace, umí pracovat s maticemi, řešit různé druhy rovnic, zobrazovat grafy funkcí. Při práci se zlomky či výrazy ukáže vždy několik tvarů výsledků, aniž bychom upřesňovali, v jakém tvaru výsledek chceme (roznásobené závorky, nebo naopak součinový tvar) (Obr. 1.4). Tedy není nutné si pamatovat spousty příkazů či anglických ekvivalentů.

Ti, kteří se angličtiny nebojí, nemusí jen počítat elementární příklady, ale také zadávat složitější příkazy. Na obrázku (Obr. 1.5) vidíme dotaz pro objem a povrch koule při poloměru 10. Na obrázku je dotaz položen celou větou. Wolfram Alpha anglickou větu správně vyhodnotí. Není nutné psát dotazy celými větami, stačí heslovitě.



Obrázek 1.5: Objem a povrch koule

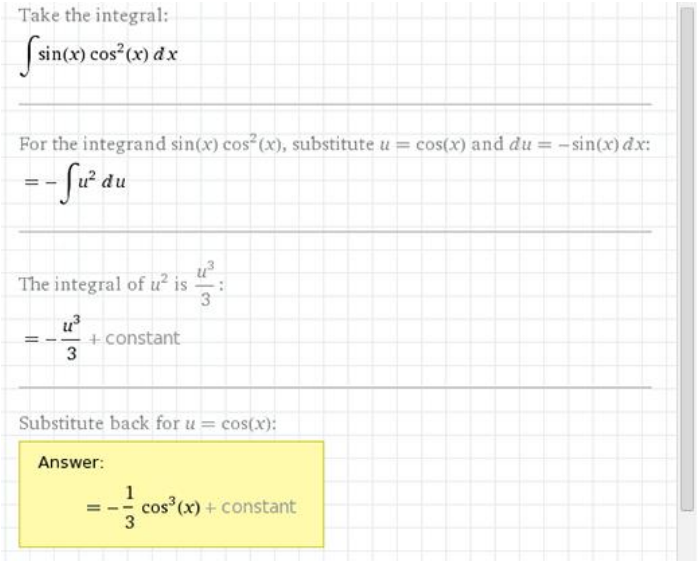


Obrázek 1.6: Vkládání obrázků

1.2.2 Další funkce Wolfram Alpha

Aplikace Wolfram Alpha nabízí další funkce (Obr. 1.6) přístupné po vytvoření účtu a následném přihlášení.

První užitečnou funkcí může být přidávání do tzv. Favorites – oblíbených. Zadaný dotaz si můžeme jakkoliv pojmenovat a kdykoliv si jej znovu vyvolat



Obrázek 1.7: Step-by-Step Solutions

a případně upravit. Může se hodit při různých typech rovnic, integrálů apod.

Další funkce je History – historie – funguje standardně. K dispozici je nahlédnutí do seznamu zadávaných dotazů a možnost je přidat mezi oblíbené, či znovu zobrazit jejich výsledky. Historii je možno kdykoliv vymazat.

Již jsem zmiňovala možnost stáhnutí a uložení výstupů a to v mnoha formátech, např. GIF, JPG, EPS, TEX, PDF, CDF, NB (formáty umožňující otevření ve Wolfram Mathematica). Wolfram Alpha umožňuje i nahrávání souborů – obrázků, grafů aj.

Užitečnou funkcí může být funkce Step-by-step Solutions (Obr. 1.7), zobrazení řešení krok za krokem. Umožňuje nám podívat se, jak je daný problém možno řešit. K dispozici máme zobrazení řešení po krocích, nebo zobrazení všech kroků najednou. Wolfram Alpha ovšem řeší vše algoritmicky s nějakými výhodami a ulehčeními, ale občas je z těchto důvodů řešení Step-by-step kostrbaté či složité. Ale i takovéto řešení může mnohdy pomoci.

1.2.3 Použití ve výuce matematiky

Aplikaci Wolfram Alpha lze využívat v hodinách matematiky. Můžeme ji používat pro kontrolu výpočtů, spolu s projektorem názorně ukazovat grafy. Díky jednoduchosti ovládání a syntaxi zadávání dotazů lze používat i bez předchozí

přípravy. Nebo naopak předpřipravit si příklady, vložit si je do oblíbených a pak používat ve vyučovacích hodinách.

Při řešení rovnic, integrálů, derivací a podobných úloh máme možnost zobrazit si řešení krok po kroku. S pomocí projektoru můžeme toto řešení využít přímo v hodině jako nápovědu, vždy ukázat jen jeden následující krok a nad dalšími kroky necháme studenty přemýšlet. Případně můžeme zobrazit celé řešení najednou pro kontrolu studentům.

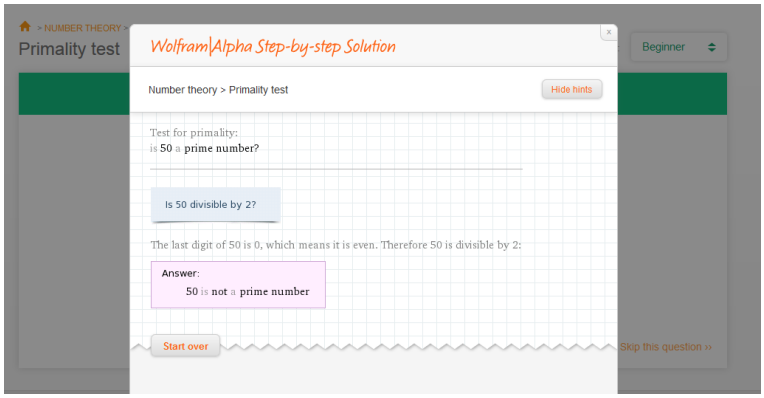
Aplikace může být využita při procvičování doma. Žák si může kontrolovat výsledky nebo i celé postupy, případně si nechat od Wolfram Alpha poradit jeden početní krok a dále se pokoušet přemýšlet sám. Této možnosti mohou využít nejen studenti základních či středních škol, ale i škol vysokých.

1.3 Wolfram Demonstration

Internetový projekt Wolfram Demonstration nabízí téměř 10 000 demonstrací z různých oborů – biologie, chemie, fyzika, umění aj. Stačí zadat slovo v angličtině a aplikace nám nabídne související demonstrace. Nebo můžeme potřebné demonstrace najít podle rozdělení do oborů. Pro spuštění demonstrací musí být v počítači nainstalovaný program CDF Player, který je zdarma a ke stažení přímo na stránce Wolfram Demonstration Projekt. V kapitole 2.4 se podíváme na Wolfram Demonstration blíže, řekneme si, jak takové demonstrace vytvořit. Představím podrobněji demonstrace, kterých jsem autorkou.

1.4 Wolfram Problem Generator

Wolfram Problem Generator je novinka webového rozhraní. Zjednodušeně by se dalo říci, že je to jednoduchá sbírka úloh. Prozatím je zaměřena jen na některá odvětví matematiky – aritmetika (základní operace s celými čísly, zlomky), teorie čísel (dělitelnost, prvočísla, největší společný dělitel, nejmenší společný násobek), algebra (výrazy, komplexní čísla, řešení rovnic, polynomy), lineární algebra (operace s vektory, skalární a vektorový součin, matice a operace s nimi), kalkulus (nespojitost funkcí, derivace, integrály), pravděpodobnost.



Obrázek 1.8: Nápovědy Wolfram Problem Generator

Podle typu matematického problému jsou úlohy řazeny vždy do dvou obtížností – začátečník a pokročilý. Obecně Wolfram Problem Generator funguje tak, že na dané téma generuje úkoly a rovnou kontroluje správnost odpovědí. Jestliže je odpověď správná, objeví se další úkol. Pokud je špatná, ukáže se řešení krok po kroku. Oproti Wolfram Aplha je tady řešení propracovanější, protože opravdu vysvětluje každý krok. Před zobrazením každého dalšího kroku řešení úlohy položí Wolfram Problem Generator návodnou otázku, aby se řešitel zamyslel a zkusil spočítat příklad sám s malou či větší pomocí (Obr.: 1.8). Ukážeme si na jednom příkladu z teorie čísel – dělitelnosti (Obr.: 1.8).

1.4.1 Použití ve výuce matematiky

Wolfram Problem Generator lze použít při domácí přípravě. Tentokrát jako generátor úloh s okamžitou zpětnou vazbou. Při problémech s učivem může pomoci zobrazení řešení Step-by-step. Vše je opět v angličtině, ale není potřebná pokročilá znalost, slovíčka si můžeme najít v internetovém překladači. Ve většině případů to ale není potřebné, protože matematické znaky v angličtině mají stejný význam jako v jazyce českém.

Wolfram Problem Generator lze použít v hodinách matematiky jako generátor úloh na dané téma. Můžeme či nemusíme využít zpětnou kontrolu. Myslím si, že ve většině případů je zbytečné používat Step-by-step řešení, protože učitel by měl být schopen tuto funkci nahradit. Tedy na hodinu je učitelům prospěšnější generátor úloh.

2. Program Wolfram Mathematica

2.1 Historie programu Wolfram Mathematica

Někteří lidé považují vytvoření programu Wolfram Mathematica za počátek moderních technických výpočtů. Cílem tvůrců programu bylo vytvořit jeden a pro všechny stejný systém, který by mohl zvládnout různé aspekty technických výpočtů. Klíčem bylo vytvořit nový druh symbolického počítačového jazyka [1].

Vznik programu Wolfram Mathematica je spojován se jménem Stephena Wolframa. První zmínka o programu jako takovém je z roku 1988, kdy přišla na svět první verze tohoto softwaru. Stephan Wolfram se se svou skupinou matematiků, fyziků a programátorů zasloužil o velký rozvoj tohoto programu [3].

Největší změnou prošel program Wolfram Mathematica v roce 2007 při verzi 6. Byly vytvořeny interaktivní části dokumentu, umožňující pracovat s matematickými, fyzikálními a dalšími daty a mnoho dalšího. Nyní, v roce 2014, je k dispozici verze Wolfram Mathematica 9.0 [2].

Z počátku se program používal především pro výpočty ve fyzikálních vědách, inženýrství a matematice. Ale v průběhu let se program stal důležitým nástrojem v pozoruhodně širokém rozsahu oborů technických, ale i jiných. Program se naučili používat přední vědci, což pomohlo k mnoha světovým objevům [2].

Největší část komunity uživatelů programu se sestává z odborníků z technických a dalších oblastí. Čím dál více se program používá ve vzdělání, nejen matematice a fyzice, ale také chemii a dalších předmětech [2].

Na technické úrovni je program Wolfram Mathematica považován za hlavní počin softwarového inženýrství. Je to jeden z největších aplikačních programů a obsahuje širokou škálu nových originálních algoritmů a významných inovací [1].

O vývoj programu se zasloužil tým pod vedením Stephena Wolframa, který vede společnost Wolfram Research. Díky jejich úspěchu vznikly i nástroje Wolfram Alpha, Wolfram Demonstration a Wolfram Problem Generator, o kterých již byla řeč.

2.2 Základní informace o softwaru

Program Wolfram Mathematica patří mezi Computer Algebra Systems (CAS). O systému bylo napsáno asi 500 knih a víc než 15 000 článků na celém světě [4].

Program představuje programový systém pro vykonávání numerických i symbolických výpočtů a vizualizaci dat. Silnou stránkou tohoto systému je samostatný programovací jazyk na bázi jazyků umělé inteligence.

Jazyk systému Wolfram Mathematica je navržený tak, že umožňuje velmi jednoduchou manipulaci s grafickými objekty. Využití možností grafického programování vede k lepší prezentaci probíraného učiva. Je možné velmi jednoduše vytvářet animace např. pro demonstrování závislostí grafu funkce na změně parametrů.

2.2.1 Struktura programu Wolfram Mathematica

Základní části, které tvoří systém Wolfram Mathematica [4, s. 4-6]:

1. Kernel, přesný název MathKernel, výpočtové jádro systému. Toto jádro vykonává všechny výpočty. Je napsáno v jazyce C a je jedním z největších matematických systémů. Základním principem jádra je realizovat všechny výpočty přesně. Při chybě ve výpočtu nemusíme restartovat celou aplikaci, stačí znovu spustit výpočetní jádro.

2. Front End je program, který zabezpečuje komunikaci jádra (Kernel) s uživatelem. Příkazy vpisujeme do tohoto prostředí, po příkazu na jejich realizaci jsou automaticky odeslané do Kernelu. Výsledek se opět zobrazí ve Front Endu. Základem tohoto prostředí jsou Notebooky, je to plně interaktivní dokument. Je rozdělen na buňky různých typů. Pomocí závorek na pravé straně můžeme jednotlivé buňky schovat či zobrazit. Základní typy buněk jsou vstupní (zde zadáváme příkazy, které se vyhodnocují stiskem Shift+Enter) a výstupní (kde se zobrazí vyhodnocení příkazů). Notebook můžeme používat jako matematický editor, můžeme realizovat úpravy a formátování textu, definovat interaktivní prvky.

3. Package – doplňují systémové knihovny, protože uživatel běžně využívá asi 20 % schopností aplikace. Je vhodné, aby knihovny, které jsou využity jen občas, zbytečně nezatěžovaly aplikaci a načetly se jen v případě potřeby.

2.3 Tvorba projektů

V programu Wolfram Mathematica lze vytvářet zajímavé výukové projekty. Program je nejen kalkulačkou, ale lze v něm psát i text. V programu lze tvořit i prezentace, této části jsem se ale nevěnovala.

Samotný program Wolfram Mathematica má interaktivní nápovědu, ke které se dostaneme pomocí nabídky v horní liště, nebo pomocí klávesy `F1`, či napsání otazníku před příkaz, který chceme v nápovědě zobrazit. Nápověda lze použít offline, tedy bez připojení k internetu. Každý příkaz má obecně vysvětleno použití, které je následně ukázáno přímo na příkladech. Celá nápověda je Notebook, kde si příkazy můžeme vyzkoušet, přepsat, zkopírovat. Na jedné stránce nápovědy je i několik ukázek použití daného příkazu i s dalšími možnostmi. Nápověda obsahuje i vysvětlení tvorby funkcí a demonstrací. Celá nápověda je v angličtině.

V dnešní době již existuje několik českých návodů. Nemělo by tedy žádný význam, popisovat zde všechny funkce. Chci ukázat, jak program funguje a co od ní můžeme očekávat.

Při práci s programem Wolfram Mathematica se vpravo objevují modré čáry, které ohraničují tzv. buňky. Buňky rozdělujeme na vstupní a výstupní. Vstupní buňky jsou ty, do kterých zadáváme příkazy, na začátku takovéto buňky je `In[]:=`. Výstupní buňky začínají `Out[]=` a v nich najdeme výsledky dotazů. Buňky umožňují další činnosti: můžeme je označovat, mazat, kopírovat, měnit jejich styl písma, grafiku, zobrazování, povolovat a zakazovat jejich vyhodnocování, slučovat je, sbalit je, aby nešly v danou chvíli vidět, přiřazovat jim nastavení pomocí stylů atd. [5]

2.3.1 Základní možnosti použití

Následující kapitola se věnuje základním možnostem použití programu Wolfram Mathematica s ohledem na výuku matematiky na základních či středních školách.

Obsahuje použití základních funkcí, které mohou najít využití nejen v hodinách matematiky.

Nejjednodušší použití programu Wolfram Mathematica je využívat jej jako kalkulačku – zadávat jednoduché početní operace $+, -, \cdot, :$. Program umožní zadat matematický výraz (např. $3 + 5$) a po stisknutí klávesové zkratky Shift + Enter vypočítá program očekávaný vysledek 8. Matematické operace se značí běžnými znaky, násobení se značí hvězdičkou (*), nebo se může znak vynechat a nahradí se mezerou. Pozor tedy na zápis `ab`, program Wolfram Mathematica tento zápis bere jako jednu proměnnou a ne násobení dvou proměnných. Správně tedy pro násobení čísel a a b musíme zapsat `a b`, nebo `a*b`.

Program Wolfram Mathematica při psaní příkazů rozlišuje velká a malá písmena, neboli je case sensitive. Každá vestavěná funkce programu začíná velkým písmenem. Argumenty funkce se dávají do hranatých závorek. Mezi běžně používané funkce v matematice, obzvlášť na základních či středních školách, patří goniometrické funkce, logaritmy, kombinační čísla. Často v hodinách matematiky potřebujeme základní konstanty. Seznam takovýchto příkazů nalezneme v tabulce 2.1.

Většina příkazů je založena na anglických ekvivalentech příslušné činnosti a poměrně jednoduše je lze dohledat v nápovědě.

Pouhé zadání rovnice a stisk Shift + Enter nevyvolá žádný výpočet či úpravu. Práce s rovnicemi vyžaduje použití zabudované funkce `Solve` (od anglického řešit). Tato funkce má dva parametry, zadanou rovnici a za čárkou proměnnou. Tedy není problém počítat rovnice s parametrem jako je vidět na příkladu `In[1]`. Upozornit musím na rozdíl mezi znaky == a =. Jedno rovnítko znamená *přiřadit* (někteří znají z programování, přiblížím v kapitole 2.3.2), dvě rovnítka v programu Wolfram Mathematica vedle sebe značí rovná se ve smyslu rovnosti dvou hodnot, proto v rovnici musíme použít ==.

Příkaz `Solve` se používá i pro řešení soustav rovnic o více neznámých. Prvním parametrem této funkce jsou všechny rovnice, které máme v soustavě, ve složených závorkách. A jako druhý parametr budou všechny proměnné opět ve složených závorkách. Na příkladu je vidět, v jakém tvaru program Wolfram Mathematica podává řešení rovnice.

`Pi`	konstant π
`I`	imaginární jednotka
`E`	Eulerovo číslo e
`Sin[60°]`	funkce sinus 60 deg
`Cos[Pi/4]`	funkce sinus $\frac{pi}{4}$
`Tan[0]`	funkce tangens 0 deg
`Log[15]`	přirozený logaritmus z 15
`Log[2,15]`	logaritmus při základu 2 z 15
`Sqrt[2]`	odmocnina ze dvou
`Abs[-9]`	absolutní hodnota čísla -9
`8!`	faktoriál čísla 8
`Binomial[9,4]`	kombinační číslo 9 nad 4

Tabulka 2.1: Tabulka základních funkcí

```
In[1]:=  Solve[{x + a y == 1, x - y == a^2}, {x, y} ]
Out[1]=  {{x -> 1 - a + a^2, y -> 1 - a}}
```

Pro práci s komplexními čísly je potřeba zmínit několik příkazů. Jak již bylo řečeno, imaginární jednotka se v progtamu Wolfram Mathematica zadává pomocí velkého tiskacího `I`. Důležité funkce pro počítání s komplexními čísly jsou:
`Re[x]`– vrátí reálnou část komplexního čísla x,
`Im[x]`– vrátí imaginární hodnotu komplexního čísla x,
`Conjugate[x]`– tato funkce spočítá komplexně sdružené číslo k číslu x.

Od obyčejného stolního kalkulátoru se program Wolfram Mathematica liší také v tom, že podá výsledek ve tvaru, jakém my chceme. Pokud požadujeme výsledek ve tvaru desetinného čísla, slouží k tomu funkce `N[]`, která má jeden či dva parametry, první je dané číslo, které chceme vyčíslit, druhý (nepovinný) parametr je počet desetinných míst, na který chceme číslo vypsat. Tato funkce se dá zapsat ještě druhým způsobem, jako je vidět na příkladě.

```
In[2]:= Sin[Pi/4]//N
Out[2]= 0.707107
```

Víme, že $\sin(\frac{\pi}{4})$ je zlomek $\frac{\sqrt{2}}{2}$, který i program Wolfram Mathematica vypočítá jako výsledek, ale pokud chceme desetinné číslo, tak za funkci sinus dáme `//N`.

Pokud potřebujeme desetinné číslo ve tvaru zlomku, poslouží nám funkce `Rationalize[]`. I tuto funkci můžeme zapsat dvěma způsoby:

```
Rationalize[0.75]
0.75//Rationalize
```

Pro práci s matematickými výrazy najdou využití tyto příkazy –`Expand[]`, `Factor[]`, `Simplify[]`. Příkaz `Expand[]` můžeme v hodinách využít pro roznásobení mnohočlenů. Opačnou funkci má příkaz `Factor[]`, který z mnohočlenu vytvoří součinový tvar. Ve výuce můžeme použít pro kontrolu úpravy výrazů. Tyto funkce jsem použila i v mé demonstraci, která generuje příklady na úpravy mnohočlenů (Obr. 2.10). Příkaz `Simplify[]` zjednodušuje matematické výrazy.

Některé funkce v programu Wolfram Mathematica umožňují práci s výrazy, které obsahují goniometrické funkce, tak, aby využívaly jejich vlastností. Pro jejich úpravy pak obdobně existují příkazy `TrigExpand[]` a `TrigFactor[]`. Tyto funkce mohou najít také uplatnění při řešení goniometrických rovnic. I tyto příkazy lze použít až za výraz pomocí `//TrigExpand, //TrigFactor`.

Již na střední škole se začíná pracovat s pojmem matice, proto je vhodné umět s maticemi pracovat i v tomto programu. Obsah matice uložíme do seznamu pomocí příkazu `List[]`, každý řádek musíme umístit do složených závorek. Pokud ale dáme na výstup pouze `List[]`, nebude seznam vypadat jako požadovaná matice. Musíme ke správnému zobrazení použít příkaz `MatrixForm[]`, který má dvě možnosti použití. První pomocí funkce s parametrem, jak vidíme v příkladě `In[4]`. Druhá možnost je napsat za samotný parametr pomocí `//MatrixForm`, vstupní kód `In[5]`:

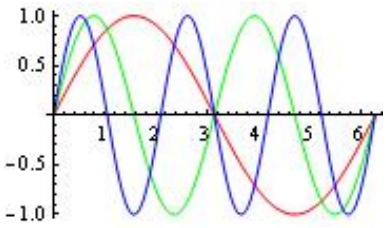
```
In[3]:= List[{1, 2, 3}, {3, 2, 1}, {4, 2, 6}]
In[4]:= MatrixForm[List[{1, 2, 3}, {3, 2, 1}, {4, 2, 6}]]
In[5]:= InList[{1, 2, 3}, {3, 2, 1}, {4, 2, 6}] //MatrixForm
Out[3]={{1, 2, 3}, {3, 2, 1}, {4, 2, 6}}
```

$$\text{Out[4]=} \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 4 & 2 & 6 \end{pmatrix}$$

Out[5]= $\begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 4 & 2 & 6 \end{pmatrix}$

Další funkcí, kterou program Wolfram Mathematica disponuje, jsou grafy, ať už grafy funkcí jedné proměnné, dvojrozměrné či trojrozměrné grafy. Každý graf se dá otáčet, zvětšit. Základní funkcí pro tvorbu grafů je příkaz `Plot[]`, zobrazí graf funkce jedné proměnné. `Plot[Sin[x], {x, 0, 2 Pi}]` zobrazí graf funkce sinus proměnné x, která nabývá hodnot od 0 do 2Π.

Pokud chceme zobrazit více grafů v jednom obrázku, stačí je přidat do složených závorek jako první parametr. Funkce `Plot[]` umožňuje spoustu dalších možností, jak obrázek grafu udělat názornější. Jednotlivé grafy mohou mít různou tloušťku, barvu. Na obrázku (Obr.: 2.1) je ukázka grafu funkce sinus se třemi různými argumenty a každý graf je zobrazen jinou barvou: $\sin(x) \rightarrow$ červený, $\sin(2x) \rightarrow$ zelený, $\sin(3x) \rightarrow$ modrý. Zdrojový kód v programu Wolfram Mathematica pak vypadá takto:



Obrázek 2.1: Graf funkcí

```
Plot[{Sin[x], Sin[2x], Sin[3x]},{x,0,2 Pi},
PlotStyle->{Red, Green, Blue}]}
```

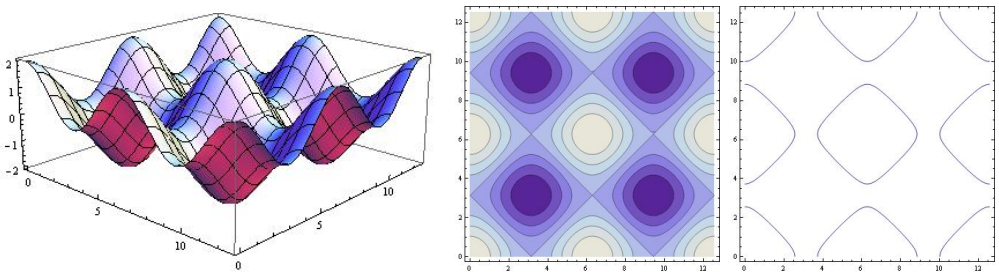
Dalšími vlastnostmi grafů, tedy funkce `Plot[]`, si již může každý prozkoumat v návodu. Zde by bylo zbytečné, abych vše podrobně popisovala.

Můžeme zobrazovat nejen grafy funkcí, ale také grafy posloupností či jiné nespojité grafy. K těmto zobrazením slouží funkce `DiscretePlot[]` či `ListPlot[]`. I tyto fungují podobně jako funkce `Plot[]` a disponují podobnými možnostmi, jen graf je složen z bodů.

Dostáváme se k funkcím více proměnných. Zůstanu u dvou proměnných a přiblížím dva typy grafů. Mezi základní příkaz grafu patří `Plot3D[]` (Obr: 2.2a), který zobrazí trojrozměrný graf. Má tři parametry: zadání grafu, minimální a maximální hodnoty proměnných. Opět nám program Wolfram Mathematica umožňuje si ho jakkoliv přizpůsobit z hlediska barev, tlouštěk a dalších možností. Program

přináší celkem dobrou vizualizaci těchto trojrozměrných grafů, jelikož se s nimi v Notebooku dá otáčet všemi směry.

Druhou zajímavou možností zobrazení grafu funkce dvou proměnných je příkaz `ContourPlot[]`, přináší pohled *zhora* (Obr: 2.2b), neboli kolmý pohled na rovinu xy. `ContourPlot[]` také umožňuje ukázat průřez libovolnou funkční hodnotou, jak je vidět na obrázku (Obr: 2.2c), kde hodnota je $\frac{1}{6}$.



(a) Plot3D (b) ContourPlot (c) ContourPlot

Obrázek 2.2: Grafy funkce $\cos(x) + \cos(y)$

Takováto zobrazení získáme po vložení následujících zdrojových kódu.
Zdrojový kód k obr.2.2a:

```
Plot3D[Cos[x] + Cos[y], {x, 0, 4 Pi}, {y, 0, 4 Pi}]
```

Zdrojový kód k obr. 2.2b:

```
ContourPlot[Cos[x] + Cos[y]==1/6, {x, 0, 4 Pi}, {y, 0, 4 Pi}]
```

Zdrojový kód k obr. 2.2c:

```
ContourPlot[Cos[x] + Cos[y], {x, 0, 4 Pi}, {y, 0, 4 Pi}]
```

Pro výuku matematiky na středních a základních školách by měly tyto příkazy stačit. S jejich znalostí můžeme spočítat hodnoty výrazů logaritmy, goniometrickými funkcemi, absolutní hodnotou, kombinačními čísly, určit kořeny lineárních, goniometrických, logaritmických či exponenciálních rovnic. Tyto funkce mohou posloužit na hodině matematiky především pro kontrolu správnosti výpočtů. Obdobně nám poslouží faktorizace výrazů, operace s maticemi, které jsme si uvedli. Ve výuce učitelé mohou ocenit zobrazení grafů funkce dle jejich požadavků.

Systém Wolfram Mathematica obsahuje asi největší kolekci zabudovaných algoritmů a funkcí pro přesné i přibližné výpočty. Mezi nejpoužívanější patří numerická řešení rovnic, hledání inverzních matic, vlastních čísel matic, problémy z teorie čísel. Těmito funkcemi se zde zabývat nebudu, jelikož jejich použití na většině středních či základních škol, je minimální. Pokud požadovaný algoritmus nenajdeme, můžeme si jej sami vytvořit. Jak, to se pokusím ukázat v následující kapitole.

2.3.2 Programujeme ve Wolfram Mathematica

V současnosti se již na středních školách studenti učí programovat. Nejčastěji se na programování pro začátečníky používá programovací jazyk Pascal, který prošel od jeho začátků velkou změnou a stálé se používá pro výuku programování ve školách. Profesionální programátoři používají jiné typy jazyků, z těch nejznámějších je to C, C++, Java, C#, Python, aj.

Programový systém Wolfram Mathematica je ve skutečnosti nejen Computer Algebra System (CAS), ale zároveň je to jeden z plnohodnotných programovacích jazyků. Občas se používá i na výuku programování. Zahrnuje v sobě totiž nejen základní procedurální programování, ale také funkcionální, rekurzivní, aj. [4]

Klasické programovací jazyky fungují většinou v módu kompilátora. To znamená, že nejdříve musíme napsat celý program, potom jej zkompilujeme (přeložíme), vytvoříme spustitelný soubor (např. `*.exe`) a až jeho spuštěním si můžeme ověřit, že program skutečně počítá. Na rozdíl od klasických programovacích jazyků pracuje program Wolfram Mathematica standardně v módu interpreta (do módu kompilátora lze přepnout). To znamená, že námi zadaný příkaz není potřebné kompilovat, vytvářet samospustitelný soubor a potom ověřovat jeho funkčnost. Interpeter znamená, že po zadání příkazu celou programovou strukturu odešleme na zpracování a interpreter (namísto kompilátora) postupně naše příkazy čte a interpretuje, tedy vykonává jednotlivé části tohoto *programu*. Tento způsob práce jsme doteď používali. Programový celek tvořila vlastně naše buňka, kterou jsme odeslali na zpracování a po její interpretaci interpreterem jsme získali výsledek [4].

Programování v programu Wolfram Mathematica je vhodné, pokud má algoritmus programu obsahovat matematické funkce, které jsou zde definovány. Poté

nám programování v programu přináší spoustu zjednodušení při práci. Na následujících stránkách si ukážeme základní příkazy používané při programování a také cykly, kterými program disponuje.

Již jsem upozorňovala na rozdíl znaků == a =. Dvě rovnítka znamenají, že hodnoty na obou stranách se mají rovnat, kdežto jedno rovnítko znamená přiřazení. Tedy si můžeme zadefinovat proměnnou a přiřadit jí hodnotu a nyní bude v této proměnné přiřazená hodnota, dokud nepřiřadíme hodnotu jinou. V následujícím příkladě jsme do proměnné a přiřadili číslo 8, v dalším příkazu při použití této proměnné se proměnná a automaticky zamění za její hodnotu.

```
In[1]:= a = 8;
a + 3
Out[1]= 11
```

Proměnným můžeme přiřazovat nejen čísla, ale i funkce, výrazy, hodnotu jiných proměnných, textový řetězec aj.

Pro vymazání hodnoty z proměnné slouží příkaz `ClearAll[]`, v dalším použití a neobsahuje hodnotu, vypíše se tedy samotná proměnná.

```
In[2]:= a = 8;
a + 3
ClearAll[a]
a + 3
Out[2]= 11
Out[3]= 3 + a
```

Vše, co jsme dosud napsali do vstupní buňky, se nám taky zobrazilo ve výstupní buňce. Toto lze ale potlačit, nemusíme zobrazovat ve výsledku vše, ale jen to požadované. K tomu nám poslouží středník, tedy znak `;`. Na následujících příkladech vidíme rozdíl výstupů.

```
Int[4]:= a = 3
Solve[x^2 + 2 x + a == 2, x]
Out[4]= 3
Out[5]={{x -> -1}, {x -> -1}}
```

```
------------------------------
In[6]:= a = 3;
Solve[x^2 + 2 x + a == 2, x]
Out[6]= {{x -> -1}, {x -> -1}}
```

U matic jsme již měli příkaz `List[]`, můžeme si seznam libovolně pojmenovat, pro ukázku si ho pojmenujeme `Seznam[]`. Důležitá je pak funkce, která vypíše hodnotu na určitém místě. Pokud máme uloženou pouze řadu čísel, tak pro zavolání pátého prvku použijeme `Seznam[[5]]`. Máme-li `Seznam[]` dvojrozměrný, například matici, a chceme zavolat hodnotu třetího řádku a druhého sloupce, napíšeme příkaz `Seznam[[3,2]]` a na výstupu dostaneme požadovanou hodnotu.

Dostáváme se k tvorbě vlastních funkcí. Každá námi vytvořená funkce má tvar `nazev[parametry s podtržítkem]=coChciVracet`, použití vidíme na příkladě `In[10]`. Chceme si uložit složitější matematický výraz a zkoumat jeho hodnoty v závislosti na dvou parametrech. Nazveme ho `ukazka` a jeho parametry jsou x, y. Všimněme si použití středníku pro nevypisování zadaného výrazu. Místo obvyklého přiřazení pomocí `=` je zde dvojice znaků `:=`. Rozdílu se podrobně věnovat nebudeme. Stačí si zapamatovat, že při definici proměnné se obvykle používá `=` a při definici funkce obvykle `:=` (u naší funkce f mimochodem fungují oba způsoby).

```
In[10]:=
 ukazka[x_, y_]:=
Sqrt[(x^2 + 3 y^3 + Cos[x Pi])/(15 - y + Sin[y Pi])];
ukazka[5, 2]
ukazka[4, 1]
ukazka[0, 0]
```

Out[10]=$4\sqrt{\frac{3}{13}}$
Out[11]= $\sqrt{\frac{10}{7}}$
Out[12]= $\frac{3}{\sqrt{15}}$

Další poměrně často používanou vestavěnou funkcí v programování je *tisk*, neboli `Print[]`. Vše, co je v hranatých závorkách této funkce, se objeví ve výstupní

buňce. Pokud je v nich proměnná, vytiskne se její hodnota, pokud chceme vytisknout text, vložíme jej do uvozovek. Použití si ukážeme u příkladu s použitím funkce `If`.

Důležitou součástí programování jsou cykly a podmíněné příkazy, které můžeme znát například z Pascalu či jiných programovacích jazyků. I program Wolfram Mathematica použití cyklů a podmíněných příkazů umožňuje. Ukážeme si, které to jsou a jak je například můžeme použít.

Nejdříve se podrobněji podíváme na podmíněné příkazy. První je `If` – pokud. Použijeme, chceme-li následující krok udělat jen v nějakém případě, za nějaké podmínky. Například pokud je číslo x menší než 0, tak do x přiřaď jeho absolutní hodnotu. V následujícím příkladě jsem uložila do hodnoty proměnné x číslo -5 a použila onu podmínku a následně nechala vypsat číslo x. Vidíme, že se hodnota čísla x opravdu změnila na jeho absolutní hodnotu.

```
In[15]:= x = -5;
If[x < 0, x = Abs[x]];
x
Out[15]= 5
```

Funkce `If` lze použít i v případě, že pokud něco platí, udělej tento krok, a pokud neplatí, udělej druhý krok. Obecně jsou dvě možnosti syntaxe `If[podmínka, příkazy oddělené středníkem]`. Nebo druhá možnost, chceme-li udělat něco jiného, pokud podmínka neplatí `If[podmínka, příkazy oddělené středníkem, alternativní příkazy oddělené středníkem]`. V parametrech nemusí být jen jeden příkaz, jak můžeme vidět na následujících příkladech, které vyhodnocují, jestli číslo uložené v `koren`, je kořenem polynomu $f = x^2 + 2x + 1$. Podmínka se ptá, zda hodnota polynomu v `koren` (-1) je rovna nule, pokud ano, vypíše se *Kořen je -1.* V druhém příkladě je v `koren` uloženo číslo 2, v takovém případě se vytiskne *2 není kořenem* a do `koren` se uloží číslo o jedna větší. Zdrojový kód těchto příkladů sám o sobě asi nenajde využití, ale mohl by být součástí nějakého algoritmu.

```
In[16]:= f[x_] := x^2 + 2 x + 1;
koren = -1;
If[f[koren] == 0, Print["Kořen je ",  koren, "."] ,
 Print[koren, " není korenem."]; koren = koren + 1]
Out[16]= Kořen je -1.
-------------------------------------------------
In[16]:= f[x_] = x^2 + 2  x + 1;
koren = 2;
If[f[koren] == 0, Print["Kořen je ",  koren, "."] ,
 Print[koren, " není kořenem."]; koren = koren + 1]
Out[16]= 2 je kořenem.
```

Druhý podmíněný příkaz je `Which[]`, funguje velmi podobně jako `If[]`. Umožňuje ovšem více podmínek a pro každou, pokud platí, následuje nějaký krok. Syntaxe je následující `Which[podmínka1, příkaz1, podmínka2, příkaz2,..., podmínkaN, příkazN]`. Toto použijeme, pokud se potřebujeme rozhodnout o následujícím kroku podle výsledku kroku předcházejícího.

Mezi příkazy pro cykly patří `For[], While[], Do[]`, které svými názvy naznačují funkčnost. Prvně si podrobněji ukážeme `For[]` cyklus. Použijeme ho tehdy, pokud chceme několikrát udělat určitou posloupnost kroků a přesně víme kolikrát to bude. Syntaxe je `For[inicializace, podmínka, zvýšení čísla, příkazy]`. Jasnější nám to bude z příkladu, což je jednoduchý algoritmus na generování určitého počtu čísel. Můžeme libovolně změnit hodnoty proměnných: `pocetcisel` – počet čísel, který chceme generovat, `maximum, minimum` – maximum a minimum značí interval, z kterého chceme generovaná čísla. Zde vidíme i novou vestavěnou funkci `RandomInteger[]`, která vrací náhodné přirozené číslo. Další použití můžeme vidět na zdrojovém kódu demonstrace (Obr. 2.10).

```
In[17]:= pocetcisel = 4;
maximum = 8;
minimum = 80;
For[i = 1, i <= pocetcisel, i++,
 Print[RandomInteger[{maximum, minimum}]]]
Out[17]= 64
```

```
Out[18]= 60
Out[19]= 55
Out[20]= 78
```

Český ekvivalent `While[]` znamená dokud, zatímco atd. Tedy příkazy v tomto cyklu jsou vykonávány, dokud platí podmínka. Parametry jsou `While[podmínka, příkazy]`. Ukážeme si jeden příklad pro lepší pochopení. Zadaná sekvence kroků spočte největší společný dělitel dvou čísel. Jako druhá ukázka kódu je vestavěný algoritmus na počítání největšího společného dělitele dvou čísel. Tento příklad je převzatý z nápovědy programu Wolfram Mathematica [6]. Vidíme použitou novou funkci `Mod[]`, která vrátí zbytek po dělení čísla a číslem b. Dosud nezmíněný znak $!=$ znamená nerovná se, je různý od.

```
In[22]:= {a, b} = {27, 6};
While[b != 0, {a, b} = {b, Mod[a, b]}];
a
Out[22]= 3
---------------------------------------
In[23]:= {a, b} = {27, 6};
GCD[a, b]
Out[23]= 3
```

Poslední cyklus je `Do[]`. Umožňuje více možností použití, ukážu pouze jednu. Používá se, pokud chceme opakovaně udělat nějaký krok a vždy poupravit proměnnou. Syntaxe `Do[příkaz, i, min, max]`. První příklad tiskne mocniny čísla n, které je z množiny $\{0, 1, 2, 3, 4\}$. Druhý příklad vytváří složený zlomek. Číslo 5 v tomto příkladě značí počet opakování. Tyto příklady můžeme nalézt v nápovědě [6]

```
In[24]:= Do[Print[n^2], {n, 0, 4}]
Out[24]= 0
Out[25]= 1
Out[26]= 4
Out[27]= 9
Out[28]= 16
```

```
-----------------------------------
```

```
In[30]:= t = x; Do[t = 1/(1 + t), 5]; t
```

$$\text{Out[30]}= \cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+x}}}}}$$

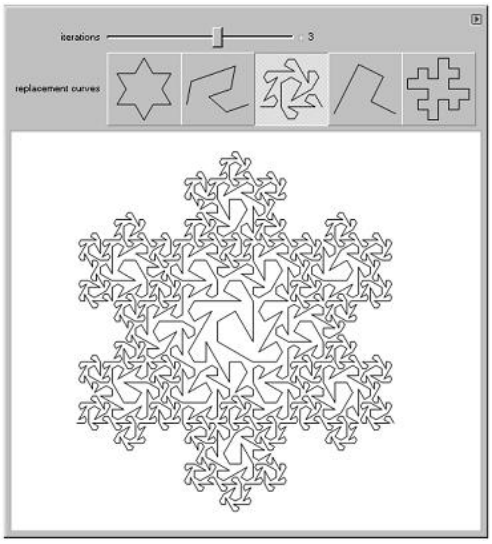
K tvorbě procedur, které neobsahují pouze jeden matematický výraz, ale více kroků, je zapotřebí znalost příkazu `Module[]`. Do hranatých závorek tohoto příkazu uložíme posloupnost kroků, které chceme s danými proměnnými provést. Volání algoritmu je pak jednodušší, nemusíme pro opětovné použití psát všechny kroky, zavoláme pouze jednu proceduru. Syntaxe je `Module[seznam lokálních proměnných, soubor příkazů]`. Takto tedy všechny příkazy uložíme do nové funkce. Např. `rozklad[pocetuloh_]:= Module[parametry, příkazy]`. Podrobnější použití této funkce ukážu až v kapitole 2.4.

Důležitá je práce s polynomy, vzhledem k tomu, že to nejsou čísla, ale řetězce, i program Wolfram Mathematica s nimi pracuje trochu jinak. Na některé odlišnosti bych ráda upozornila. Zatím jsme se setkali se znaky `==`, `!=`. U polynomů se ovšem musí použít znaky tyto `===`, `=!=`. První totiž porovnává pouze čísla, tyto již řetězce. Tedy pro hodnoty polynomů používáme první sadu, pro samotné polynomy druhou sadu znaků.

2.4 Demonstrace

V první části práce jsem se zmínila o Wolfram Demonstration Project, internetový nástroj firmy Wolfram Research dostupný z internetového prohlížeče. Jak jsem již napsala v kapitole 1.3, pomocí programu Wolfram Mathematica můžeme tyto demonstrace vytvářet. Rozhodla jsem se této části věnovat nejvíce. Mohla jsem vytvářet různé applety a projekty v programu Wolfram Mathematica, ale ty by byly přístupné pouze pro uživatele programu Wolfram Mathematica. Kdežto všechny demonstrace mohou být zveřejněné na internetu a k jejich prohlížení pak stačí mít nainstalovaný CDF Player, který je zdarma. Po nainstalování programu CDF Player do počítače si může každý mé demonstrace nejen prohlížet, ale i využívat je v hodinách matematiky.

Vzhledem k tomu, že na stránkách Wolfram Demonstration je již spousta prací, nebylo jednoduché vybrat si témata pro tvorbu nových. Pomohla mi v tom vlastní zkušenost z vyučování matematiky na základní škole. Nejdříve se podíváme na to, co to vlastně ty demonstrace jsou.



Obrázek 2.3: Demonstrace

Jak již bylo řečeno, na stránkách `http://demonstrations.wolfram.com` je již nepřeberné množství různých zajímavých programů, které znázorňují nejrůznější problémy. Na obrázku (Obr.: 2.3) můžeme jednu zajímavou demonstraci z oblasti rekreační matematiky vidět. Demonstrace umožňuje zobrazit jednu z pěti fraktálních křivek zobrazených na tlačítkách (Obr. 2.3). Fraktální křivky jsou generovány z počáteční křivky (často pravidelný mnohoúhelník) a jedné nebo více reprodukčních křivek. Opakovaně, každá linie je nahrazena kopií ve správném měřítku jedné z reprodukčních křivek. Tlačítka ukazují první iteraci (opakování) použité reprodukční křivky. Kroky, které můžeme měnit, znamenají počet iterací. Pokud chceme vidět původní křivku, musíme nastavit krok na 0 [8].

V následující kapitole si ukážeme, jak se projekty tvoří a následně popíši ty, které jsem sama vytvořila.

2.4.1 Tvorba demonstrací ve Wolfram Mathematica

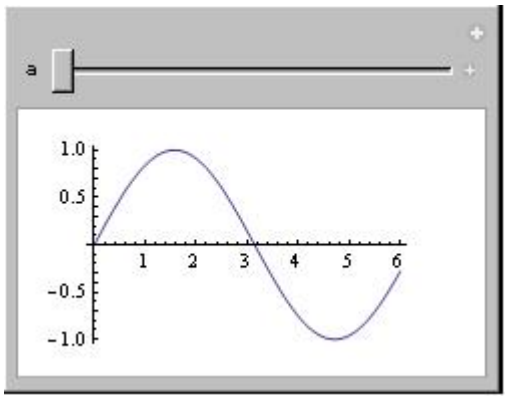
Pomocí výukového videa jsem se naučila tvořit demonstrace, tato část není složitá a přichází až na úplný závěr [8]. Předchází jí ovšem tvorba daného kódu, která se provádí v již známém prostředí programu Wolfram Mathematica. Pokud máme vymyšlené téma, algoritmus a povedlo se nám jej přepsat do řeči programu Wolfram Mathematica, použijeme zmiňovanou funkci `Module[]` a celý algoritmus vložíme do jedné funkce.

Nyní můžeme přímo v Notebooku vytvořit vizuální část projektu. K tomu nám poslouží `Manipulate[]`. Její parametry jsou: funkce, jejíž výsledky se mají zobrazovat a interval, z kterého je parametr, na němž je funkce závislá. Tedy jedno z jednodušších použití.

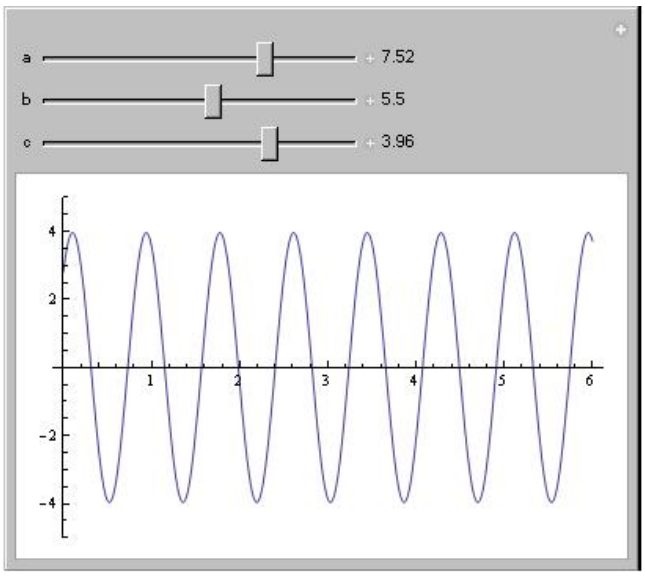
```
Manipulate[Plot[Sin[x (1 + a x)], {x, 0, 6}], {a, 0, 2}]
```

Výsledek vidíme na obrázku 2.4.

Obdobně můžeme psát místo grafu funkce sinus graf libovolné jiné funkce. Můžeme mít i více parametrů, kterým se bude měnit hodnota (Obr.: 2.5). Parametry si můžeme posouvat sami, nebo pomocí plus zapnout automatický měnič parametrů. Jak je později popsáno u obrázku 2.17.



Obrázek 2.4: Manipulate sinus

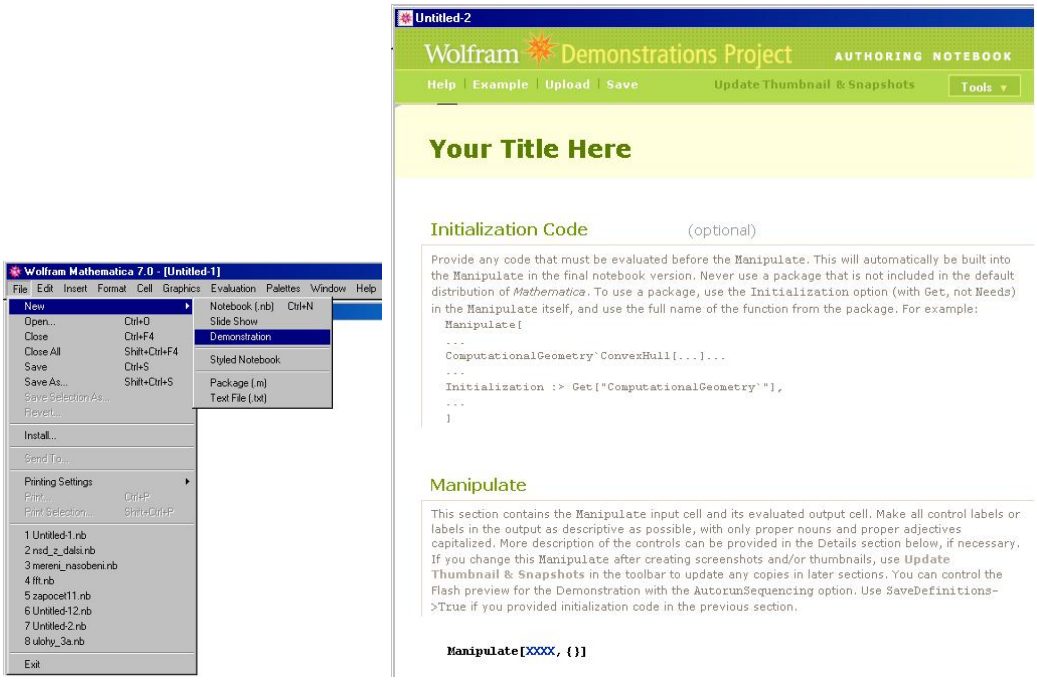


Obrázek 2.5: Graf funkce sinus

K tvorbě podobných prací existuje velmi málo českých textů. Na internetu je možné shlédnout několik videí, kde autor tvoří podobný projekt, ale bohužel většina videí není ani v angličitně. Ale i přesto se z videa dá něco naučit. Proto se pokusím popsat nejdůležitější možnosti `Manipulate`, podrobnosti již uživatel může najít v nápovědě. Nejprve budu způsoby využití funkce `Manipulate[]` vysvětlovat následujících demonstracích, ve kterých jsem již vícero možnosti této funkce použila, pak doplním ty možnosti, které jsem sice nepoužila, ale můžou být pro čtenáře užitečné.

Ukážeme si, jak bychom hotovou práci (Obr. 2.5) umístili na internet. Velmi podobná práce již na stránkách je od jiného autora, tedy jsem vložení nedokončila.

Při vytvoření nového souboru v programu Wolfram Mathematica nepoužijeme možnost Notebook, ale přímo demonstration (Obr. 2.6). Otevře se nám šablona pro tvorbu projektu. Změníme `Your Title Here` na vlastní text, v našem případě `Graf funkce cosinus`.



Obrázek 2.6: Otevření projektu

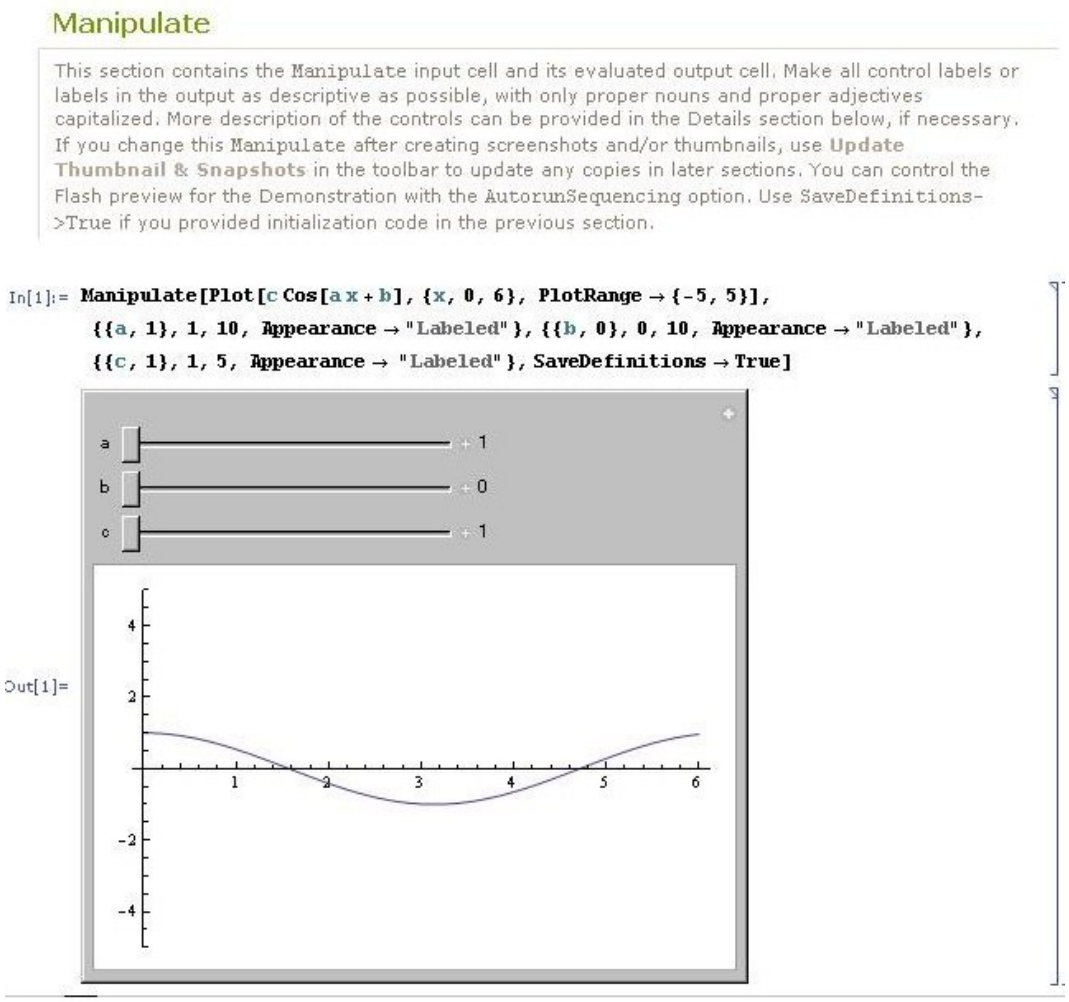
Do sekce `Initialization code` vložíme funkci, kterou zobrazujeme, v našem případě je to `Plot[c Cos[a x + b],x,0,6, PlotRange -> -5, 5]`. Další sekce `Manipulate`, zde vložíme celý kód naší práce.

```
Manipulate[
 Plot[c Cos[a x + b], {x, 0, 6}, PlotRange -> {-5, 5}],
  {{a, 1}, 1, 10, Appearance -> "Labeled"},
  {{b, 0}, 0, 10, Appearance -> "Labeled"},
  {{c, 1}, 1, 5, Appearance -> "Labeled"},
  SaveDefinitions -> True].
```

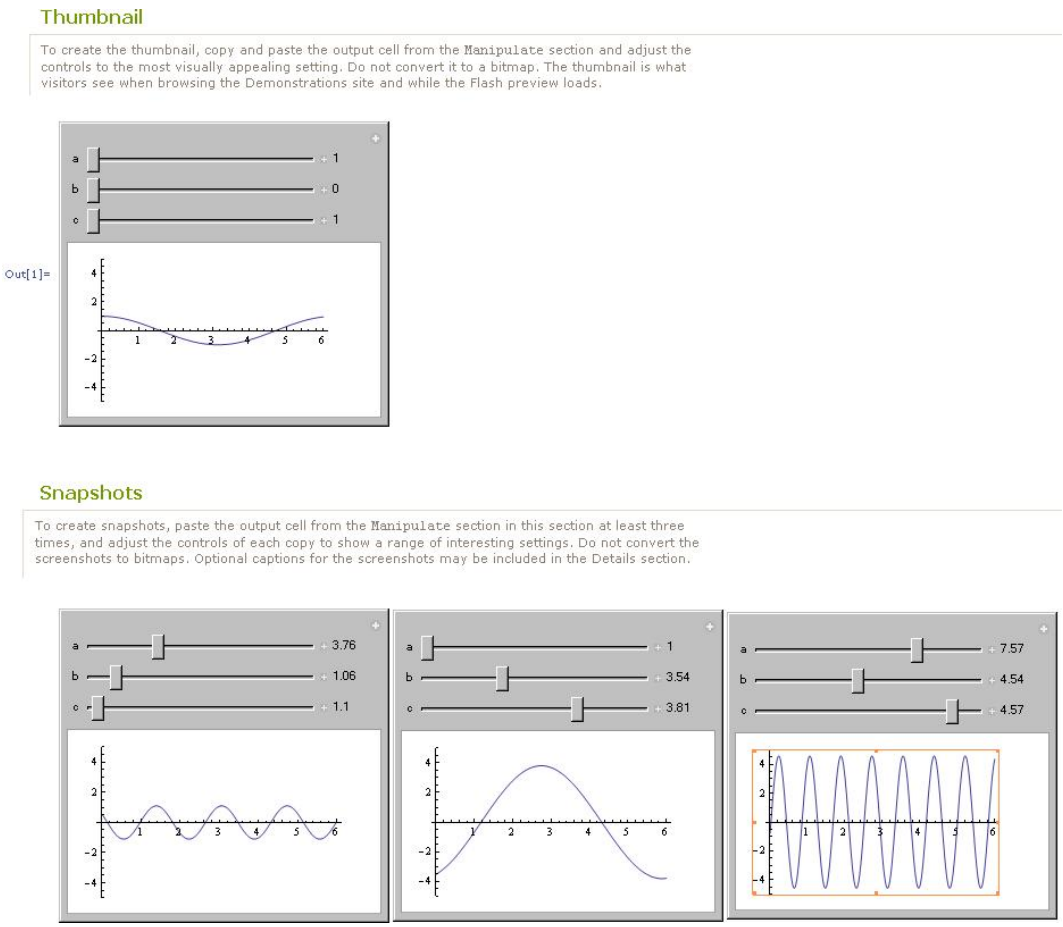
Vidíme části zdrojového kódu, se kterými jsme se ještě nesetkali. První je `PlotRange->`, tímto definujeme obor hodnot, který se bude na grafu stále zobrazovat. Všimněme si, že parametry jsou definovány trochu jinak, než jsme si v úvodu říkali. Kód `{a, 1}` udává, na jaké hodnotě parametr začíná, další čísla jsou minimální a maximální hodnota parametru. Kdyby bylo za čárkou třetí číslo, určovalo by hodnotu krokování parametru, neboli o kolik bude větší další hodnota. Pokud tento údaj chybí, tak si ho program Wolfram Mathematica nastaví sám. Spojení `Appearance -> "Labeled"` říká, že hodnota parametru se zobrazí na konci pohyblivé lišty. A poslední neznámé `SaveDefinitions -> True` ukládá definici funkce, aby se po otevření nemusela práce inicializovat a neukazovala něco nevyžádaného. Vložili jsme kód do části `Manipulate` a nyní ho musíme pomocí Shift + Enter inicializovat. Objeví se nám známá tabulka (Obr. 2.7).

Do části `Caption` patří stručný popis aplikace. Do sekce `Thumbnail` zkopírujeme naši vygenerovanou aplikaci a nastavíme hodnotu parametru tak, jak potřebujeme, aby tento obrázek viděli ostatní. V `Snapshots` budou tři kopie naší aplikace, pokaždé s jinými hodnotami parametrů, aby ostatní uživatelé měli náhled, co náš projekt demonstruje, aniž by ho museli spouštět (Obr. 2.8).

Do dalších částí patří hesla, která pomůžou uživateli najít naši demonstraci. Nezapomeneme na `Authoring Information`, kam napíšeme své jméno. Nyní už jen chybí umístit demonstraci na internet. Na obrázku (Obr. 2.6) vidíme v zelené liště menu, které obsahuje čtyři možnosti: `Help, Example, Upload, Save`. Pokud jsme soubor uložili, vložíme ho na stránky pomocí možnosti `Upload`. Dostaneme se na stránku Wolfram Demonstration Project, kde nás vyzvou k přihlášení. Pokud ještě nemáme účet, vytvoříme si jej. Po přihlášení se objeví stránka pro vybrání souboru, najdeme soubor s požadovanou demonstrací a klikneme na možnost `Upload`.



Obrázek 2.7: Sekce Manipulate



Obrázek 2.8: Thumbnail a Snapshots

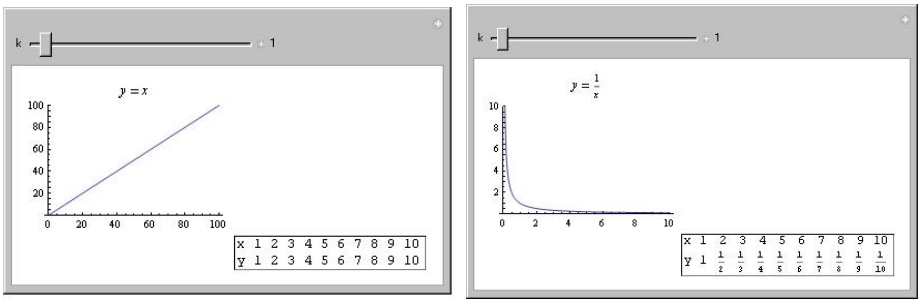
Po nahrání zkontrolujeme, zda je vše v pořádku a předložíme demonstraci k zveřejnění pomocí `Submit for publication`. Pokud nahrání proběhlo úspěšně, přijde oznamovací email, který také obsahuje URL adresu našeho projektu. Jestliže z hlediska editorů Wolfram Demonstration Project není něco v pořádku, přijde nám mail s upozorněním. Upozorňují například na opakované demonstrace, na české znaky, pro které v této době nemají podporu fontů, na špatnou velikost demonstrace aj. První vložená demonstrace od nového autora, by měla být v angličtině. Po několika publikovaných demonstracích může autor vkládat demonstrace v jiných jazycích. Práci můžeme ze stránek kdykoliv smazat. Obdobně můžeme nahrát jakýkoliv náš projekt.

Demonstrace, které na stránkách Wolfram Demonstration Project nalezneme, si můžeme stáhnout, abychom je případně mohli používat bez internetu. Některé zdrojové kódy jsou autory skryty a nemůžeme se z nich tedy učit, nebo je použít či upravit. Větší množství demonstrací zdrojové kódy zobrazuje.

Další část bude o mých pracích, ať už demonstracích, které mohou být vloženy na internet, tak také o ostatních projektech, které jsou použitelné v hodinách, ale nejsou ve formě demonstrací, ale pouze uloženy v Notebooku. To proto, že spousta demonstrací na internetu je, ale my si je nemůžeme upravovat, kvůli skrytým zdrojovým kódům, a tedy jsem některé práce nechala pouze v Notebooku, aby si případný uživatel mohl kód podle sebe upravit, případně se z něj něco nového naučit. Na těchto pracích si ukážeme použití dalších funkcí, kterými program Wolfram Mathematica disponuje. Rovnou si i řekneme, jak se dají tyto práce použít při výuce matematiky.

První dvojicí demonstrací, kterou si představíme, je Přímá úměrnost a Nepřímá úměrnost [10]. Cílem této demonstrace je žákům ukázat změny grafů při různých koeficientech úměrností. Demonstrace zobrazuje nejen graf, ale i tabulku pro několik menších hodnot (Obr. 2.9a a 2.9b). Žáci tak mají možnost si lépe představit, co se děje s grafem a hodnotami v tabulce při změně koeficientu, neboli co znamenají ty záhadné formule, které se obvykle učí či najdou v učebnicích: *kolikrát se zvětší(zmenší) jedna veličina, tolikrát se zvětší (zmenší) druhá veličina* u přímé úměrnosti a *kolikrát se zvětší (zmenší) jedna veličina, tolikrát se zmenší (zvětší) druhá veličina* u úměrnosti nepřímé.

Tato dvojice demonstrací je vhodná pro použití v hodinách matematiky při vysvětlování učiva. Případně také později při procvičování učiva na příkladech, kdy při kontrole správnosti výpočtů můžeme využít tuto demonstraci, protože si koeficient můžeme dle potřeby nastavit a tedy poté jen promítnout řešení pro kontrolu žáků, nemusíme rýsovat graf na tabuli.



(a) Přímá úměrnost (b) Nepřímá úměrnost

Obrázek 2.9: Demonstrace úměrnosti

```
Manipulate [Grid[{
   {Plot[y = k x, {x, 0, 100}, PlotRange -> {0, 100},
     PlotLabel -> Row[{Style["y", Italic], " = ", k x}]]},
   { "", "x", 1, 2, 3, 4, 5, 6, 7, 8, 9, 10},
   {"", "y", 1 k, 2 k, 3 k, 4 k, 5 k, 6 k, 7 k, 8 k, 9 k, 10 k}},
  Frame -> {None, None, {{2, 3}, {2, 12}} -> True}},
   Alignment -> Right],
 {{k, 1}, 0, 20, 0.1, Appearance -> "Labeled"}]
 ------------------------------------------------------------
 Manipulate [Grid[{
   {Plot[y = k/x, {x, 0, 10}, PlotRange -> {0, 10},
     PlotLabel -> Row[{Style["y", Italic], " = ", k/x}]]},
   { "", "x", 1, 2, 3, 4, 5, 6, 7, 8, 9, 10},
   {"", "y", k/1, k/2, k/3, k/4, k/5, k/6, k/7, k/8, k/9, k/10}},
  Frame -> {None, None, {{2, 3}, {2, 12}} -> True}},
   Alignment -> Right],
 {{k, 1}, 0, 30, 0.5, Appearance -> "Labeled"}]
```

Ve zdrojovém kódu této dvojice demonstrací vidíme některé nové příkazy. Hned v úvodu příkaz `Grid[]`, ten zobrazuje výsledky v tabulce, tedy pomocí něj jsem dosáhla celkem názorného zobrazení grafu a tabulky. Tento příkaz má v každé složené závorce, které jsou odděleny čárkou, obsah jednoho řádku. V každém řádku jsou hodnoty také odděleny čárkou. Výhodou je, že nemusíme hned při tvorbě prvního řádku specifikovat, kolik sloupců budeme potřebovat.

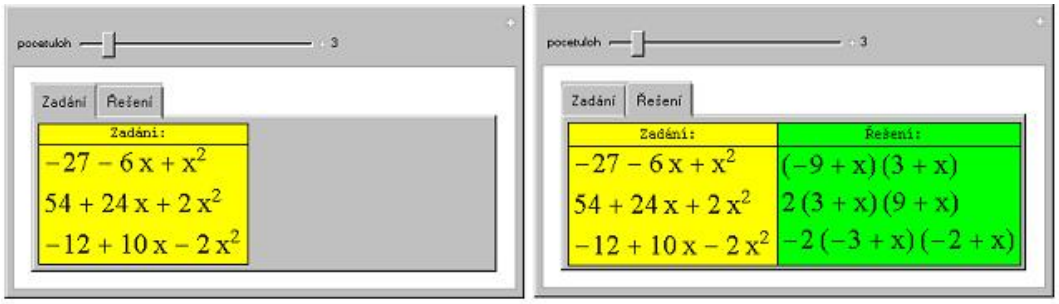
Do prvního řádku jsem vložila pouze graf a nepřemýšlela jsem nad tím, kolik dalších sloupců budu potřebovat. V dalších řádcích jsem již vložila do řádku více buněk, tedy ve výsledku má tabulka více sloupců. Potřebovala jsem pod grafem mít v následujících řádcích prázdné buňky, což jsem zajistila vytisknutím prázdného textového řetězce pomocí `""`. V dalších buňkách jsou již hodnoty, za $1k, 2k, \ldots$ si sám dosadí výsledek podle aktuální hodnoty proměnné k. Dalším zde použitým parametrem příkazu `Grid[]` vidíme `Frame`, tento parametr zobrazí čáry v tabulce, podle potřeby. `Alignment -> Right` zajistí zarovnání vpravo v rámci buněk. Toť k příkazu `Grid[]`, neboli zobrazení tabulky. Vidíme i nějaké novinky ve funkci zobrazující graf – `Plot[]`– což je zobrazení předpisu grafu pomocí `PlotLabel -> Row[]`.

Mé další dvě demonstrace jsou si velmi podobné. Obě jsou především pro procvičování problematiky úpravy výrazů. Obě demonstrace mají dvě záložky, zadání a řešení. Jedna demonstrace má v zadání rozložený polynom a v řešení jeho součinový tvar. Druhá demonstrace má řešení a zadání přesně naopak. Obě jsou vlastně generátory úloh na výrazy. Algoritmy jsem vymyslela tak, aby každý výraz bylo možné převést na součinový tvar. Každá demonstrace obsahuje dvě funkce.

```
rozklad[pocetuloh_] := Module[{pocet = pocetuloh},
   dolnimezvyberu = -10; hornimezvyberu = 10;
   linkoef = {-5, -4, -3, -2, -1, 1, 2, 3, 4, 5};
   zadani = Table[Null, {k, 1, pocet}] ;
   For[i = 1, i~<= pocet, i++,
    akoef = RandomChoice[linkoef];
    bkoef = RandomInteger[{dolnimezvyberu, hornimezvyberu}];
    ckoef = RandomChoice[linkoef];
```

```
    dkoef = RandomInteger[{dolnimezvyberu, hornimezvyberu}];
    typyulohy = RandomChoice[{0, 1, 2, 3}];
    vyraz[x_] := Expand[akoef (x + bkoef) (x + ckoef)];
    If[typulohy == 0,
     vyraz[x_] := Expand[akoef (x + bkoef) (x + ckoef)],
     If[typulohy == 1,
      vyraz[x_] := Expand[akoef (x^2 + bkoef) (x + ckoef)],
      If[typulohy == 2,
       vyraz[x_] := Expand[akoef (x^2 + bkoef) (x^2 + ckoef)],
       vyraz[x_] := Expand[akoef x^4 (x + ckoef)]]]];
    zadani[[i]] = Expand[vyraz[x]];];
   Grid[{{"Zadání: "}, {Text[Style[Column[zadani], Large]]} },
    Background -> {{Yellow}}, Frame -> All]];
-------------------------------------------------------------
rozkladreseni[pocetuloh_] :=
 Grid[{{"Zadání: ", "Řešení: "},
  {Text[Style[Column[zadani], Large]],
    Text[Style[Column[Factor[zadani]], Large]]},
  Background -> {{Yellow, Green}}, Frame -> All]
```

Zakladní funkce této demonstrace se jmenuje `rozklad[]` a má jeden parametr, kterým je počet úloh, který chceme zobrazit. Vidíme podrobnější použití již zmiňované funkce `Module[]`. Ve funkci používáme několik proměnných, ze kterých pak pomocí funkcí `RandomChoice[]` (náhodný výběr z několika čísel) a `RandomInteger[]` (náhodný výběr celého čísla z daného intervalu) vybíráme čísla, která jsou koeficienty tvořených výrazů. Proměnná `zadani` je tabulkou, neboli seznamem, abychom do ní mohli ukládat daný počet zadání. Následně vidíme použití `For` cyklu, protože známe přesný počet kroků, který je uložen v proměnné `pocet` a jeho hodnota je počet zadání, který chceme zobrazit. Ve `For` cyklu se tedy vytvoří daný počet zadání. V každém opakování se vygenerují nové koeficienty výrazu a vybere se typ úlohy pomocí podmíněného příkazu `If[]`. Máme čtyři typy úloh podle toho, jaké závorky se násobí, jaké mocniny čísla x obsahují. Toto si samozřejmě každý může ve zdrojovém kódu upravit podle své potřeby a potřeby



Obrázek 2.10: Demonstrace roznásobení

žáků. Po vygenerování výrazu (`vyraz[x_]`) se expandované (roznásobené) zadání uloží do příslušné pozice seznamu `zadani`. Po vytvoření daného počtu zadání použijeme opět příkaz `Grid[]`, který nám zabezpečí zobrazení do tabulky.

Druhá funkce `rozkladreseni[]` pouze vytvoří tabulku se zadáními a příslušnými řešeními. Používá seznam, který se v první funkci `rozklad[]` inicializoval. Do sloupečku zadání se vypíší pouze zadání a do vedlejšího sloupečku vypíšeme řešení tak, že použijeme příkaz `Factor[]` na seznam `zadani`. Aby se nám zadání a řešení zobrazovalo dostatečně velké a pěkně spolu na odpovídající řádek, musela jsem změnit výrazy na textové řetězce a zvětšit jejich písmo, toto všechno pomocí `Text[Style[Column[zadani], Large]]`. Zabarvení pozadí dosáhneme pomocí nepovinného parametru `Background ->`.

Ukázali jsme si funkci pro rozklad na součinový tvar, pro opačný postup je funkce stejná, jen jsou prohozeny příkazy `Expand[]` a `Factor[]`. Abychom dostali demonstraci použijeme příkaz `Manipulate[]`.

```
Manipulate[ TabView[{"Zadání" -> rozklad[pocetuloh],
   "Řešení" -> rozkladreseni[pocetuloh]}],
   {{pocetuloh, 2}, 1, 20, 1, Appearance -> "Labeled"},
    SaveDefinitions -> True]
```

Vidíme opět novou funkci, kterou je `TabView`. Tento příkaz vytváří záložky a v každé zobrazuje danou funkci. V uvozovkách je název záložky a za šipkou daná funkce. V našem případě máme dvě záložky, ale může jich být libovolné množství. Vše ostatní ve zdrojovém kódu již známe. Na obrázcích (Obr. 2.10) vidíme obě záložky této demonstrace pro čtyři zadání. Musím upozornit, že při každé změně počtu zadání se původní zadání přegenerují.

Ve vyučování můžeme promítnout námi zvolený počet zadání, řešení schovat libovolným otevřeným oknem a nechat žáky pracovat a následně řešení odkrýt, aby si mohli zkontrolovat své výpočty. Nebo můžeme demonstraci doporučit žákům na domácí procvičování, protože demonstrace bude zveřejněna na internetu [10].

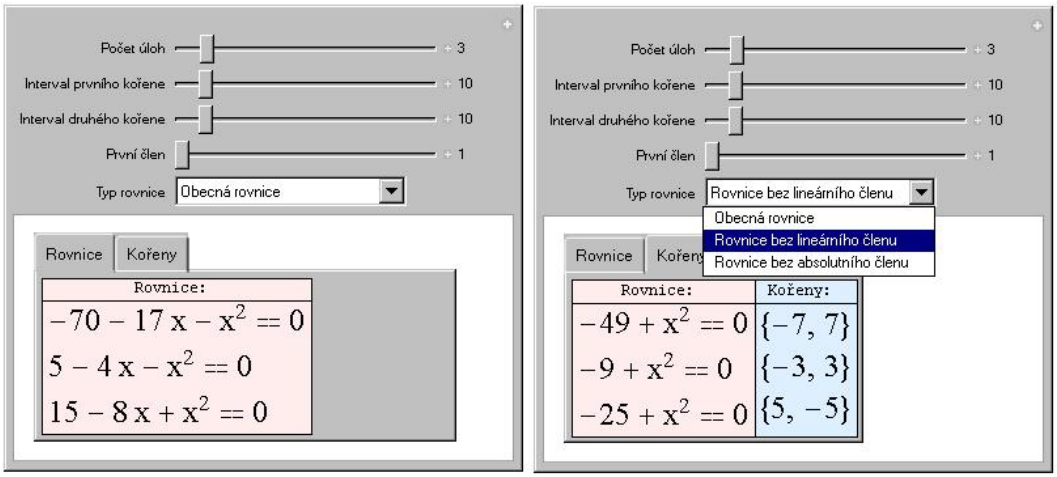
Demonstrace *Generování kvadratických rovnic* patří také mezi generátory úloh. Vygeneruje kvadratickou rovnici s celočíselnými kořeny a umožňuje následně kořeny zobrazit pro kontrolu. Aplikace nabízí možnost vybrat si typ rovnice (obecná rovnice, rovnice bez lineárního členu, rovnice bez absolutního členu). Základem jsou dvě funkce – `generator[]`, `reseni[]`.

```
generator[pocetuloh_, int1_, int2_, prvni_, cislo_] :=
  Module[{pocet = pocetuloh, max1 = int1, max2 = int2,
    a = prvni, typ = cislo},
   koreny = Table[Null, {l, 1, pocet}];
   zadani = Table[Null, {k, 1, pocet}] ;
   For[i = 1, i <= pocet, i++,
    If[typ == 1, koren1 = RandomInteger[{-max1, max1}];
     While[koren1 == 0,
      koren1 = RandomInteger[{-max1, max1}]];
     koren2 = RandomInteger[{-max2, max2}];
     While[koren2 == 0  ||  koren2 == -koren1,
      koren2 = RandomInteger[{-max1, max1}]];
     koreny[[i]] = {koren1, koren2};
     akoef = RandomInteger[{-a, a}];
     While[akoef == 0,
      akoef = RandomInteger[{-a, a}]];
     rovnice[x_] = akoef (x - koren1) (x - koren2) == 0;
     zadani[[i]] = rovnice[x] // Expand,
     If[typ == 2, koren1 = RandomInteger[{-max1, max1}];
       While[koren1 == 0,
        koren1 = RandomInteger[{-max1, max1}]];
       koren2 = -koren1;
```

```
        koreny[[i]] = {koren1, koren2};
        akoef = RandomInteger[{-a, a}];
        While[akoef == 0, akoef = RandomInteger[{-a, a}]];
        rovnice[x_] = akoef (x - koren1) (x - koren2) == 0;
        zadani[[i]] = rovnice[x] // Expand,
        koren1 = RandomInteger[{-max1, max1}];
        While[koren1 == 0,
         koren1 = RandomInteger[{-max1, max1}]];
        koren2 = 0;
        koreny[[i]] = {koren1, koren2};
        akoef = RandomInteger[{-a, a}];
        While[akoef == 0, akoef = RandomInteger[{-a, a}]];
        rovnice[x_] = akoef (x - koren1) (x - koren2) == 0;
        zadani[[i]] = rovnice[x] // Expand];
     ]];
  Grid[{{"Rovnice: "}, {Text[Style[Column[zadani], Large]]}},
    Background -> {{LightPink, Blue}}, Frame -> All ]];
-------------------------------------------------------------
reseni[pocetuloh_] :=
 Grid[{{"Rovnice = 0: ",
    "Kořeny: "}, {Text[Style[Column[zadani], Large]],
    Text[Style[Column[koreny], Large]]} },
  Background -> {{LightPink, LightBlue}}, Frame -> All];
```

Fungují velmi podobně jako funkce v předchozích demonstracích na faktorizaci a roznásobení mnohočlenů. Funkce `generator[]` má pět parametrů. Můžeme totiž měnit nejen počet zobrazovaných úloh, ale také intervaly, ze kterých jsou pomocí náhodného výběru (funkce `RandomInteger[]`) vybírány kořeny rovnice. Nejdříve jsou totiž vygenerovány kořeny rovnice a koeficient a u x^2 a následně pomocí vztahu $a(x - x_1)(x - x_2)$ vygenerováno zadání a uloženo do proměnné `zadani` na `i`-tou pozici. Kořeny rovnice se také uloží do proměnné `koreny` na `i`-tou pozici.

Proměnná `cislo` může nabývat tří hodnot z množiny $1, 2, 3$. Každá hodnota



Obrázek 2.11: Demonstrace – generování kvadratických rovnic

reprezentuje jeden typ rovnice. Hodnota 1 znamená obecnou kvadratickou rovnici, tedy nenulové kořeny x_1, x_2, pro které platí $x_1 \neq -x_2$. Hodnota 2 reprezentuje rovnici bez lineárního člen, tedy takovou, která má nenulové kořeny, pro které také platí $x_1 = -x_2$. Poslední hodnota 3 reprezentuje rovnici bez absolutního členu, taková má jeden kořen nulový. Podle uložené hodnoty v proměnné `cislo` se provede část podmínky `If`. Aby v každém typu rovnice byly pouze ty správné typy, obsahuje zdrojový kód `While` cykly, které kontrolují nenulovost kořenů a nerovnost opačných hodnot. Pokud rovnost nastane, vygenerují se nové hodnoty kořenů, které se opět zkontrolují. Při tvorbě obecných kvadratických rovnic kontroluje program nejdřív nenulovost prvního kořene a následně kontroluje druhý kořen, pro který nemají platit dvě podmínky, které jsou ve zdrojovém kódu spojeny symbolem `||`. Tento symbol znamená logickou spojku `nebo`. Pokud tedy platí jedna z podmínek, vygeneruje se nová hodnota. Zdrojový kód také kontroluje, zda koeficient a u x^2 není roven nule, v tomto případě by totiž rovnice byla nulová. Toto zajišťuje také `While` cyklus.

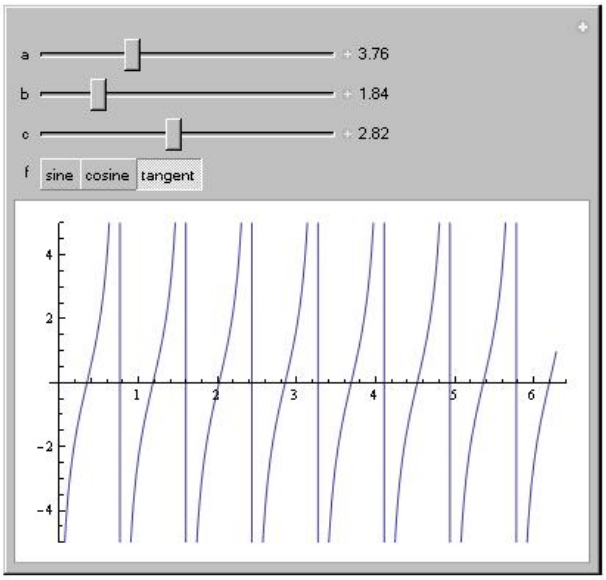
Funkce `reseni[]` již slouží pouze ke zobrazení kořenů a používá hodnoty spočtené funkcí `generator`. Pro vytvoření demonstrace musíme tyto funkce vložit do `Manipulate[]`.

```
Manipulate[
 TabView[{"Rovnice" -> generator[pocetuloh,
    int1, int2, prvni, cislo],
```

```
   "Kořeny" -> reseni[pocetuloh]}],
 {{pocetuloh, 2, "Počet úloh"}, 1, 20, 1,
  Appearance -> "Labeled"},
 {{int1, 10, "Interval prvního kořene"}, 0, 100, 1,
  Appearance -> "Labeled"},
 {{int2, 10, "Interval druhého kořene"}, 0, 100, 1,
  Appearance -> "Labeled"},
 {{prvni, 1, "První člen"}, 1, 100, 1,
  Appearance -> "Labeled"},
 {{cislo, 1, "Typ rovnice"},
  {1 -> "Obecná rovnice",
   2 -> "Rovnice bez lineárního členu",
   3 -> "Rovnice bez absolutního členu"},
    ControlType -> PopupMenu}]
```

Opět jsem použila pro zobrazení zadání zvlášť záložky, tedy příkaz `TabView[]`. Tato demonstrace umožňuje měnit počet zobrazených zadání a následných řešení a intervaly, ze kterých se berou kořeny a koeficient a u x^2. Pokud je pro interval prvního kořene vybráno číslo 5, pak celý interval, ze kterého náhodně program Wolfram Mathematica vybírá je $< -5; 5 >$. Obdobně ostatní intervaly. Na obrázku (Obr. 2.11) vidíme, že u posuvníků již nejsou pouhé názvy proměnných, ale již slovní pojmenování. Ve zdrojovém kódu vidíme, kam musíme název posuvníku umístit, aby se nám správně zobrazil.

Na obrázku (Obr. 2.11) vidíme také rozbalovací menu, které nabízí na výběr typ kvadratické rovnice. Tedy určuje hodnotu proměnné `cislo`. Zdrojový kód je podobný jako u posuvníku, nejdříve název proměnné, hodnota, defaultní hodnota, název, který se zobrazí v demonstraci. Následně musíme vypsat všechny hodnoty i s jejich názvy, které uživatel uvidí. Rozbalovacího menu dosáhneme pomocí příkazu `ControlType -> PopupMenu`.



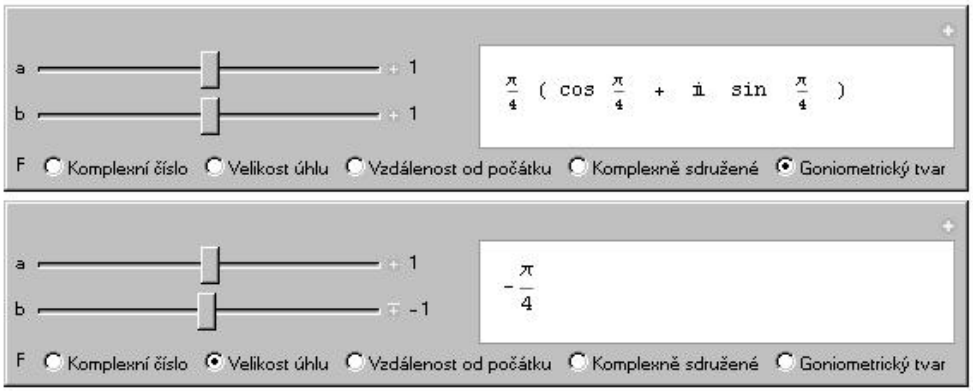
Obrázek 2.12: Goniometrické funkce

V příloze jsou zdrojové kódy a obrázky aplikací funkcí sinus, cosinus, tangens a cotangens. Pro každou funkci je jiná aplikace, u které můžeme měnit tři parametry funkce. Ukážu ale aplikaci, která obsahuje všechny tyto graf, ale ukazuje vždy právě jeden vybraný graf (Obr. 2.12). Je zde totiž použita další zajímavá možnost funkce `Manipulate[]`.

```
Manipulate[Plot[c f[a x + b], {x, 0, 2 Pi}, PlotRange -> {-5, 5}],
  {{a, 1}, 1, 10, Appearance -> "Labeled"},
  {{b, 0}, 0, 10, Appearance -> "Labeled"},
  {{c, 1}, 1, 5, Appearance -> "Labeled"},
  {f, {Sin -> "sine", Cos -> "cosine", Tan -> "tangent"}}]
```

První část zdrojového kódu již byla zmíněna, zobrazuje graf nějaké funkce f závislé na parametrech a, b, c, jejichž hodnoty jsou následně definovány. Poslední řádek je nový a vytvoří tlačítka v aplikaci. Funkci f nahrazujeme třemi jinými funkcemi podle názvu tlačítka – sinus, cosinus, tangens. Obdobně bychom si mohli udělat aplikaci pro jiné funkce, které chceme porovnávat a přiblížit tak více jejich grafy studentům.

Vytvořila jsem jednoduchou aplikaci na základní problematiku komplexních čísel. V aplikaci si můžeme zobrazit komplexní číslo $a + bi$. Hodnoty čísel a a b jsou z intervalu $< -100, 100 >$, což je možno ve zdrojovém kódu změnit. Aplikace



Obrázek 2.13: Komplexní čísla

umožňuje zobrazit samotné komplexní číslo, velikost úhlu, který svírá průvodič s osou x, absolutní hodnotu, neboli vzdálenost od počátku, číslo komplexně sdružené a goniometrický tvar tohoto komplexního čísla.

K demonstraci je potřeba pomocná funkce, kterou jsou nazvala `GonioTvar[]`, umožňuje výpis goniometrického tvaru komplexního čísla, protože funkce, která by tento tvar vypsala, v programu Wolfram Mathematica neobsahuje. Podíváme se na zdrojový kód aplikace.

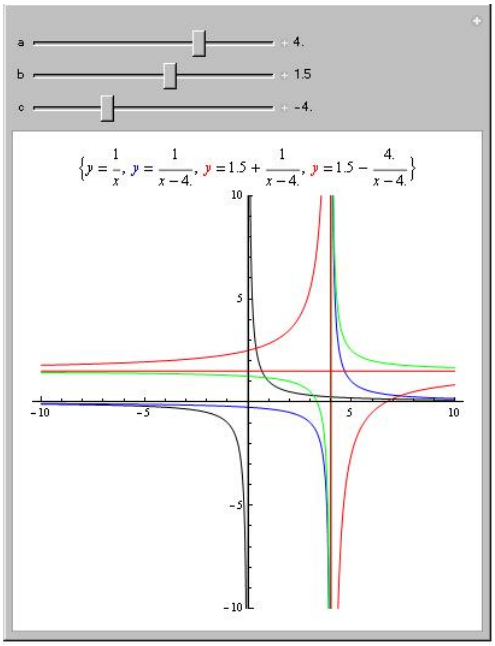
```
GonioTvar[x_] :=
  Grid[{{Arg[x], "( cos", Arg[x], " + ",
   I, " sin ", Arg[x], " )"}}];
Manipulate[
 F[a + b I], {{a, 1}, -100, 100, Appearance -> "Labeled"},
 {{b, 1}, -100, 100, Appearance -> "Labeled"},
 {F, {Simplify -> "Komplexní číslo",
   Arg -> "Velikost úhlu", Abs -> "Vzdálenost od počátku",
   Conjugate -> "Komplexně sdružené",
   GonioTvar -> "Goniometrický tvar"},
   ControlType -> RadioButtonBar,
   ControlPlacement -> Bottom},
 ControlPlacement -> Left,
 SaveDefinitions -> True]
```

Opět vidíme ve zdrojovém kódu nahrazování funkce F jinými funkcemi. Všechny funkce jsme si již představovali. Na obrázku (Obr. 2.13) vidíme zaškrtávací možnosti pro zobrazení, které vytvoří příkaz `ControlType -> RadioButtonBar`. Setkáváme se s novým rozložením polí demonstrace. Bílé pole, ve kterém jsou zobrazeny výsledky, se zobrazuje vpravo. Zaškrtávací políčka nejsou vedle bílého pole, ale pod ním. Umístění zaškrtávacích políček je zajištěno příkazem `ControlPlacement -> Bottom`, kde místo `Bottom` (dole) může být: `Top` (nahoře), `Left` či `Right` (vlevo a vpravo). Vzhledem k tomu, že určujeme pouze pozici zaškrtávacích políček, musí být tento příkaz vložen v závorce, kde políčka definujeme. Umístění bílého políčka určíme umístěním posuvníků podobným příkazem, jen v jiné části zdrojového kódu – `ControlPlacement -> Left`. Neboli posuvníky jsou vlevo a tedy bílé zobrazovací políčko se musí umístit vpravo. Mohli bychom opět místo `Left` vložit jiné umístění dle našeho požadavku.

Tuto demonstraci můžeme využít pro ukázku při probírání dané problematiky. Vhodná je pro kontrolu příkladů na převádění mezi základním tvarem $a + bi$ a goniometrickým tvarem $|z|(\cos\alpha + \imath \sin\alpha)$. Případně ji mohou studenti využít doma k procvičování absolutní hodnoty komplexního čísla, komplexně sdruženého tvaru čísla.

Rozhodla jsem se vytvořit demonstraci na téma sestrojení grafu lomené funkce. Demonstrace ukazuje fáze sestrojování grafu lomené funkce. Tedy základní $\frac{1}{x}$ (tento graf má černou barvu), graf posunutý ve směru osy x o hodnotu a, tedy $\frac{1}{x-a}$ (označen zelenou barvou), graf posunutý ve směru osy y o hodnotu b, tedy $b + \frac{1}{x-a}$ (tento graf je modrý) a následně výsledný graf $b + \frac{c}{x-a}$, který je zobrazen červeně (Obr. 2.14).

Tento applet lze použít pro ukázku tvorby grafů lineárních lomených funkcí v hodinách matematiky, kdy ukazujeme postupně změny, které způsobují jednotlivá čísla, až se dostaneme k výslednému grafu. Vzhledem k tomu, že máme volitelně nastavitelné tři parametry tohoto grafu, můžeme ukázat libovolný příklad. Pokud by nám nestačilo rozmezí parametrů v intervalu $< -10, 10 >$, můžeme si ho v počítači upravit. Následně může být tato demonstrace použita pro pro kontrolu řešení příkladu, který žáci v hodině vypracují sami. Vzhledem k tomu, že vidí všechny postupné grafy, mohou si sami lépe najít chybu. Případně opět může



Obrázek 2.14: Lineární lomená funkce

demonstrace posloužit k domácímu procvičování a zopakování látky s následnou kontrolou.

Ve zdrojovém kódu je jen jedna neznámá věc, kterou je `AspectRatio -> Automatic`, tento parametr se snaží co nejlépe přizpůsobit zobrazovaný graf, a to viditelnost os a velikost celého grafu.

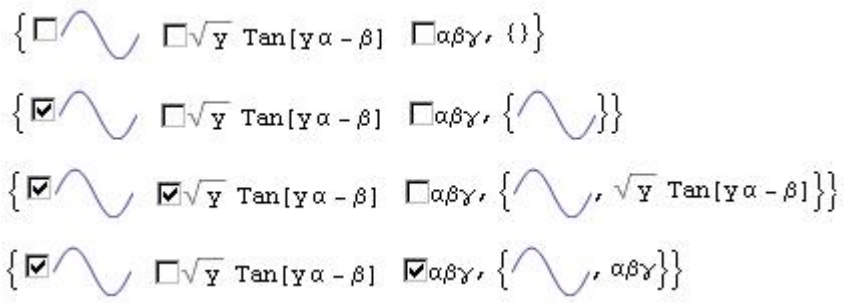
```
Manipulate[
 Plot[{1/x, 1/(x - a), b + 1/(x - a), b + c/(x - a), b},
  {x, -10, 10}, PlotRange -> {-10, 10},
  PlotStyle -> {Black, Green, Blue, Red, Red},
  AspectRatio -> Automatic,
  PlotLabel -> {Row[{Style["y", Italic, Black], " = ", 1/x}],
   Row[{Style["y", Italic, Green], " = ", 1/(x - a)}],
   Row[{Style["y", Italic, Blue], " = ", b + 1/(x - a)}],
   Row[{Style["y", Italic, Red], " = ", b + c/(x - a)}]}],
  {{a, 0}, -10, 10, Appearance -> "Labeled"},
  {{b, 0}, -10, 10, Appearance -> "Labeled"},
  {{c, 0}, -10, 10, Appearance -> "Labeled"}]
```

Chtěla bych ještě upozornit na některé zajímavé možnosti, které program Wolfram Mathematica a funkce `Manipulate[]` přináší. Je spoustu možností, co můžeme pomocí `Manipulate[]` zobrazit, ukázali jsme si, že můžeme používat a vytvářet záložky nebo tlačítky vybírat z nabízených možností. Pouze zmíním, jaké možnosti funkce ještě poskytuje. Pokud by se nějaký čtenář rozhodl použití vyzkoušet, může dohledat podrobnosti již v manuálu. Se znalostmi, které nyní máme, to nebude problém.

V okně vytvořeném pomocí `Manipulate[]` můžeme vytvořit obyčejné tlačítko, po jehož zmáčknutí provede program požadovanou akci. Bohužel nespustí jinou funkci, snad toto bude umět v nějaké příští verzi programu Wolfram Mathematica. Takové tlačítko vyvoláme pomocí příkazu `Button[]`. Další zajímavou možností zlepšit si aplikaci je tzv. `PopupMenu[]`, neboli otevírací nabídka pro výběr hodnot, se kterou jsme se setkali v demonstraci *Generování kvadratických rovnic* (Obr. 2.11). Tyto možnosti jsou vhodné, pokud chceme jen určité malé množství hodnot, pro které se má funkce počítat a tedy nechceme využít posouvací lištu.

Obrázek 2.15: Tlačítka

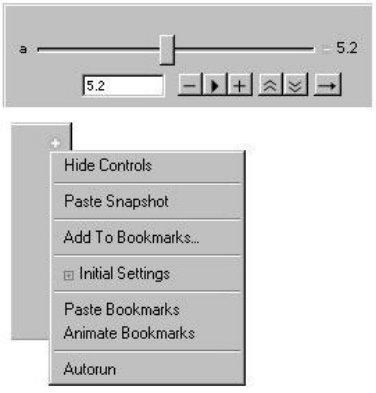
Podobně pro výběr hodnoty nějaké proměnné můžeme použít kulatá tlačítka, neboli `RadioButtonBar[]` (Obr. 2.15). Tato možnost již byla použita v demonstraci na komplexní čísla (Obr. 2.13). V těchto možnostech jsme zatím mohli vybrat pouze jednu hodnotu, program Wolfram Mathematica disponuje funkcí pro výběr více možností. Jsou to známá zaškrtávací políčka, tzv. `CheckboxBar[]`. S těmito funkcemi je často nutná znalost nového příkazu `Dynamic[]`. Představuje objekt, který zobrazuje aktuální hodnotu proměnné, která se mění. Na následujícím příkladě je zobrazeno použití `CheckboxBar[]` a následně i příkazu `Dynamic[]`. Nejdříve jsme do tří proměnných g, f, s uložili požadované funkce. Do proměnné g je to graf funkce sinus, v proměnné f je uložen výraz, který se vytiskne a v poslední proměnné jsou to pouze tři řecká písmena. Na obrázku (Obr. 2.4.1) vidíme, že se zobrazovaná hodnota na konci zaškrtávacích políček mění podle typu zaškrtnutých políček [6].



Obrázek 2.16: Zaškrtávací políčka

```
g = Plot[Sin[x], {x, 0, 2 \Pi},
 Axes -> False, ImageSize -> 40];
f = Sqrt[y] Tan[Alpha y - Beta];
s = "Alpha Beta Gamma";
-----------------------------------------------
{CheckboxBar[Dynamic[x], {g, f, s}], Dynamic[x]}
```

V neposlední řadě musím k demonstracím zmínit tzv. Autorun funkci. Doposud jsme parametry v demonstracích posouvali sami. Již byla zmíněna existence znamének plus vedle posuvníku vpravo, případně v pravém horním rohu celé demonstrace. Po kliknutí na malá znaménka plus se zobrazí lišta (Obr. 2.17), v bílém okénku s číslem můžeme ručně zadat požadovanou hodnotu parametru. Vedle jsou tlačítka, která umožňují spustit automatický běh parametru, můžeme ho vrátit zpět či posunout dopředu. Dalšími tlačítky můžeme změnit rychlost, zpomalit či zrychlit. Poslední tlačítko v této řadě změní směr běhu parametru. Po kliknutí na znaménko plus v pravém horním rohu celé demonstrace se zobrazí nabídka (Obr. 2.17). Pro nás je nejdůležitější první a poslední možnost, tedy `Hide Controls` a `Autorun`. První možnost v nabídce zapříčiní skrytí části demonstrace, ve které můžeme provádět změny hodnot parametrů. Možnost `Autorun` sama



Obrázek 2.17: Autorun

spustí automatické změny hodnot parametrů. Občas je tato možnost vhodná a dobře prezentuje, co má demonstrace předvést, občas spuštění automatického běhu není úplně šťastná volba. Ale za vyzkoušení určitě stojí.

V programu Wolfram Mathematica můžeme vytvářet i geometrické tvary, jako jsou kružnice, čtverce, trojúhelníky a další. Pokud ovšem chceme žákům tyto objekty přiblížit, si myslím, že je přirozenější používat jiný program, např. volně dostupnou Geogebru, Cabri aj. Software Wolfram Mathematica je především vhodný pro počítání a vizualizaci grafů, tvorbu demonstrací a případně programování.

2.4.2 Něco navíc

Jako poslední program jsem se rozhodla vytvořit Euklidův algoritmus pro polynomy. Není to aplikace či demonstrace jaké jsme si ukazovali doteď. Je to program přímo v Notebooku. Funkce, která spočte největší společný dělitel dvou polynomů jedné proměnné. Program taky na výstupu zobrazuje jednotlivé zbytky po dělení (posloupnost polynomiálních zbytků), aby si mohl student své řešení lépe zkontrolovat.

Program by se dal zlepšit tak, aby upravoval koeficienty polynomiálních zbytků, které rychle rostou. Úprava algoritmu, jejíž použití je vhodné, využívá jistých znalostí z algebry a eukleidovských oborů. Díky těmto znalostem bychom tedy mohli upravovat koeficienty polynomů a tak by nedocházelo k velkému růstu koeficientů. Počítačově nejméně náročná je metoda, která se nazývá Redukovaná PRS. Na tuto metodu přišel kolem roku 1850 pan Sylvester. Více se můžeme o této problematice dočíst v knize Davida Stanovského [9]. V příloze se můžeme na program, který tuto metodu používá podívat, je ovšem pro polynomy dvou proměnných.

Funkce má dva argumenty, kterými jsou polynomy, jejichž největší společný dělitel chceme vypočítat. Můžeme buďto přímo zadat dané koeficienty, nebo si je nechat vygenerovat. Na generování polynomů je v programu několik možností. Pokud použijeme nejobecnější generování polynomů, s největší pravděpodobností bude jejich společný dělitel roven 1. Z tohoto důvodu je v programu uvedena další možnost, která využívá roznásobení závorek, kde generujeme kořeny polynomu

z malého intervalu, tedy se pravděpodobnost nějakého společného dělitele různého od nuly zvýší.

Druhý program, který příloha obsahuje, je Eukleidův algoritmus pro polynomy dvou proměnných.

Tyto mé programy zde zmiňuji hlavně pro zájemce o danou problematiku a jako ochutnávku něčeho složitějšího.

3. Nejen budoucnost programu Wolfram Mathematica

3.1 Odkud čerpat další nápady

Program Wolfram Mathematica také umožňuje skrytí zdrojového kódu. Toto se hodí, pokud umisťujeme na internet své programy a nechceme, aby je kdokoliv používal, učil se z nich, případně neúmyslně změnil. Mým cílem je naopak to, aby mé programy našly široké využití u učitelů i žáků a případně i zájemců z řad široké veřejnosti. A aby se ze zdrojového kódu projektů mohli poučit či něco nového naučit. Pokud by ovšem přeci jen někdo měl zájem tuto možnost využít, popsal ji ve své prezentaci Jaroslav Reichl, na jehož stránkách je spoustu zajímavých programů do hodin matematiky, rozdělených do jednotlivých oborů a témat [11]. Všechny jeho práce mají ovšem skrytý zdrojový kód, tedy se z nich nic nenaučíme, ale stále je to může být dobrá příprava a pomoc do vyučovacích hodin.

Dalším dobrým zdrojem nápadů v češtině je http://www.mathematica.cz/, kde najdeme příklady do vyučování nejen matematiky. Ovšem tyto stránky přinášejí ještě něco víc, např. informace o licencích. Někteří uživatelé progarmu Wolfram Mathematica se pravidelně scházejí a předávají si zkušenosti a nápady, na těchto stránkách se dočteme, kdy se taková setkání konají.

3.2 Co přináší nové verze programu Wolfram Mathematica

Všechny své demonstrace a ostatní programy jsem vytvářela v programu Wolfram Mathematica 7.0. V dnešní době (rok 2014) je již k dispozici verze 9. Proto se podíváme, co nového nám přináší, na co se noví uživatelé, kteří se třeba i na základě mé práci rozhodli používat program Wolfram Mathematica, mohou těšit.

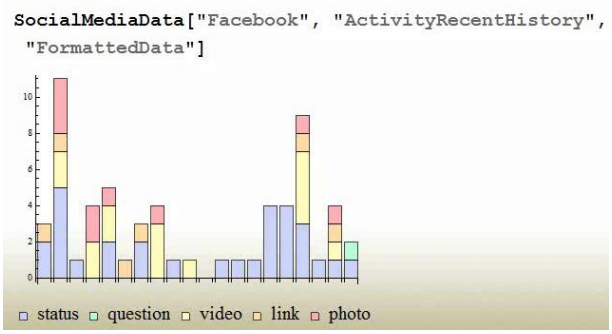
Obrázek 3.1: Nabídka pro další zdrojový kód

Hlavním přínosem nové verze je, že program Wolfram Mathematica je více interaktivní a pomáhá při tvorbě zdrojového kódu (Obr.: 3.1). Například, pokud chceme 100 náhodných čísel mezi 10 a 20, již můžeme požadavek zadat slovně v anglickém jazyce a ne nutně příkazem, jako tomu bylo např. ve verzi 7.0. Tedy zadáme příkaz `100 Random numbers between 10 and 20` a zmáčkneme Shift + Enter. Program Wolfram Mathematica příkaz vyhodnotí stejně, jako bychom zadali `RandomReal[10,20,100]`. Poté nám nabídne možnosti, které jsou k dospozici pro práci s daným výsledkem, např. seřadit, vytvořit graf či histogram, ... Po provedení následujícího příkazu opět nabídne možnosti. Pří zobrazení grafu nabídne změnu stylu, barvy, a software sám vytvoří potřebný zdrojový kód, který nám samozřejmě ukáže a uloží jej do Notebooku.

Starší verze programu Wolfram Mathematica uměly vložit obrázky do Notebooku, (pomocí příkazu `Image[]`) nejnovější verze 9 umí vložit i trojrozměrné obrázky vytvořené v nějakém jiném programu a otáčet s nimi, můžeme se tedy na ně podívat ze všech stran. Což by se nemuselo zdát potřebné, ale program Wolfram Mathematica s takovými obrázky umí i pracovat, upravovat je z technického či vědeckého hlediska. Zájemci si mohou tuto problematiku prostudovat v nápovědě.

Jako zajímavost do hodin matematiky můžeme použít další novinku nejnovější verze, kterou je zobrazování dat ze sociálních sítí, jako je například Facebook, G+, aj. Můžeme zobrazovat např. grafy počtů *lajků, komentářů*, grafy přátel (Obr.: 3.2).

Nyní již umí software pracovat i s jednotkami, převádět je, sčítat hodnoty v různých jednotkách, jako např. 2cm + 3m = 302cm. Ve starších verzích by znaky pro jednotky byly brány pouze jako názvy proměnných, teď už program



Obrázek 3.2: Graf přátel z Facebooku

Wolfram Mathematica uvažuje i o možnosti, že tyto značky mohou znamenat i něco jiného.

Rozšířila se i možnost vizualizací. Program Wolfram Mathematica má nyní zabudováno několik obrázků např. hodin, ampérmetru, voltmetru, na kterých můžeme demonstrovat matematické či fyzikální jevy. Pro zájemce o prezentace tvořené v programu Wolfram Mathematica připravili vývojáři výběr z osmi šablon.

Uvedené příklady samozřejmě tvoří jen malou část toho, co program Wolfram Mathematica 9 přináší. Najdeme v ní spoustu nových funkcí pro vyšší matematiku. Vybrala jsem to zajímavější, co se dá použít ve výuce na střední či základní škole. Nejnovější verze přináší spoustu novinek a hlavně zjednodušení pro uživatele a tedy více možností, jak dětem výuku matematiky zpříjemnit a umožnit jim lépe porozumět dané problematice.

3.3 Použití ve výuce – shrnutí

Během celého textu jsem se snažila ukázat, jak vhodně použít jednotlivé programy ve výuce. Program tedy můžeme používat jako základní početní nástroj, kterou míváme na stole. Toto je vhodné, pokud zrovna v programu Wolfram Mathematica pracujeme, ale je asi z takových důvodů zbytečné program zapínat. Zajímavější použití je již pro výpočty rovnic a úpravy výrazů. Můžeme program použít pro kontrolu výpočtů přímo v hodině, či pro kontrolu samotnými žáky při domácí přípravě.

Užitečná je možnost ukázek grafů je možnost rychlé generování grafů zadaných parametrů. Nám, učitelům, zabere příprava relativně krátkou dobu (samozřejmě časem a zkušenostmi se práce s programem Wolfram Mathematica zrychluje) a vytvořený graf je přesný a umožňuje nám grafickou úpravu, která nám bude nejvíce vyhovovat. Pro mě nejvíc smysluplně využitelné použití v hodinách, je využití oněch demonstrací, které si můžeme sami tvořit. A pokud si nevíme rady s tvorbou demonstrací, případně již nemáme čas vytvořit svou vlastní, můžeme se pokusit je najít v nabídce přímo na stránkách Wolfram Demonstration Project a s velikou pravděpodobností tam nějakou vhodnou demonstraci najdeme.

Závěr

Cílem práce bylo nejen ukázat příklady využití programu Wolfram Mathematica ve vyučovacích hodinách matematiky, ale také přiblížit práci v tomto programu především učitelům matematiky. Snažila jsem se přiblížit základní možnosti programu Wolfram Mathematica formou návodu, který ukazuje vybrané funkce. Použití funkcí jsem ukázala na příkladech. Velkou část jsem věnovala tvorbě demonstrací. Pomocí obrázků jsem provedla tvorbou od počátku až po konečné vložení demonstrace na internet. Demonstracím jsem věnovala nejvíc času, aby vzniklo povědomí o jejich existenci, protože zveřejněné práce může na internetu shlédnout každý, ať je tím každým učitel, zvídavý žák či dokonce široká veřejnost. Vytvořila jsem demonstrace, které již mohou být použity při výuce. Učitelé je tak mohou mít jako základ pro tvorbu svých vlastních prací.

Rozhodla jsem se přiblížit nejen možnosti programu Wolfram Mathematica, ale také jemu velmi blízké internetové aplikace Wolfram Alpha, Wolfram Demonstration Project a Wolfram Problem Generator. Těmto internetovým aplikacím jsem se věnovala hlavně proto, že jejich použití je zdarma. Zaměřila jsem se především na to, jak může učitel tyto aplikace použít ve výuce matematiky.

Přesto, že jsem již před psaním této práce program Wolfram Mathematica používala, tato práce mne obohatila o mnoho nových znalostí tohoto software. Uměla jsem základní příkazy a již jsem uměla některé úlohy naprogramovat. Zcela nová pro mě byla tvorba demonstrací. Nebyla to lehká část, protože anglické texty pro mě nejsou jednoduché a v češtině toho moc o demonstracích v programu Wolfram Mathematica nenajdeme. I proto jsem pojala práci částečně jako návod. O existenci Wolfram Demonstration Project jsem se dověděla náhodou a jsem ráda, že jsem se tomuto nástroji a tvorbě demonstrací ve své práci věnovala a tak přispěla především k rozšíření svých znalostí, ale i možnosti využití programu kolegy učiteli na školách.

Eventuální návazná práce na tuto bakalářskou práci by se mohla více věnovat tvorbě dalších programů, ať už demonstrací či jiných. A případně zjistit, jak na tyto programy reagují žáci ve výuce.

Seznam použité literatury

[1] *Wolfram Research*. Wolfram Mathematica – History [online]. [cit. 2014-02-22]. Dostupné z: http://www.wolfram.com/company/mathematica-history.html.

[2] *Wolfram Research*. Wolfram Mathematica – Scrapbook [online]. [cit. 2014-02-22]. Dostupné z: http://www.wolfram.com/company/scrapbook/.

[3] Stephan Wolfram. *The Life and Times of Stephen Wolfram* [online]. [cit. 2014-02-22]. Dostupné z: http://www.stephanwolfram.com/scrapbook/.

[4] J. Dobrakovová, M. Kováčová, V. Záhonová. *Mathematica 5.2 pre stredoškolských učiteľov*. Študijné materiály , STU Bratislava, 2006.

[5] Musil, Ladislav. *Programování v Mathematice: Přednášky 1-2*. 2011 [online]. [cit. 2014-02-22]. Dostupné z: https://edux.fit.cvut.cz/oppa/BI-PMA/prednasky/Pr1-2.pdf.

[6] *Wolfram Research*. Tutorial Wolfram Mathematica. Dostupné z: http://reference.wolfram.com/mathematica/guide/Mathematica.html.

[7] Slavík, Antonín. *Mathematica pro začátečníky* [online]. [cit. 2014-02-22]. Dostupné z: http://www.karlin.mff.cuni.cz/ slavik/info.html.

[8] *Wolfram Demonstration Project*. How To Create a Demonstration [online]. [cit. 2014-02-22]. Dostupné z: http://reference.wolfram.com/mathematica/howto/CreateADemonstration.html.

[9] Stanovský, David. *Základy algebry*. 1. vyd. Matfyzpress, Praha, 2010.

[10] *Wolfram Demonstration Project* [online]. [cit. 2014-03-20]. Dostupné z: http://demonstration.wolfram.com.

[11] Reichl, Jaroslav. *Mathematica – fórum* [online]. [cit. 2014-03-08]. Dostupné z: http://www.mathematica-forum.cz/.

[12] Elkan, spol. s r.o.. *Mathematica Elkan* [online]. [cit 2014-03-20]. Dostupné z: http://www.mathematica.cz/.

Příloha

Příloha obsahuje všechny programy a demonstrace vytvořené v programu Wolfram Mathematica (s příponou *nb*), o kterých jsme se v práci zmínila. Každá demosntrace má dva soubory. Jeden je obyčejný program s aplikací s českými popisky, druhý soubor je ve formě demonstrace s anglickými titulky, tento soubor je možno vložit na internet. Součástí CD je i instalační soubor programu CDF Player.

`Seznam souborů na CD:`

Moznosti_vyuziti_programu_WM_ve_vyuce_matematiky.pdf

Složka *Demonstrace*:

PrimaUmernost-demonstrace.nb

NeprimaUmernost-demonstrace.nb

LomenaFunkce-demonstrace.nb

RoznasobeniMnohoclenu-demonstrace.nb

RozkladMnohoclenu-demonstrace.nb

Složka *Programy*:

PrimaUmernost.nb

NeprimaUmernost.nb

LinearniLomenaFunkce.nb

RoznasobeniMnohoclenu.nb

RozkladMnohoclenu.nb

GoniometrickeFunkce.nb

KvadratickeRovnice.nb

EukleiduvAlgoritmus2promenne.nb

EukleiduvAlgoritmus1promenne.nb

Složka *Instalace*:

CDFPlayer_9.0.1_WIN.exe