



INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Moderní přístup k aplikaci matematických dovedností v přírodovědných a ekonomických oborech  
reg. č.: CZ.1.07/2.2.00/28.0168

## Software Mathematica pro geografy

Miloš Fňukal, David Smrčka, Petr Kladivo

Olomouc 2015

Oponenti: PhDr. Šárka Brychtová, CSc.  
Mgr. Miloslav Šerý

Neoprávněné užití tohoto díla je porušením autorských práv a může zakládat občanskoprávní, správněprávní popř. trestněprávní odpovědnost.

Tato publikace neprošla redakční jazykovou úpravou.

© Miloš Fňukal, 2015

© Univerzita Palackého v Olomouci, 2015

Olomouc 2015

1. vydání

ISBN 978-80-244-4472-7

## Obsah

<b>1. Úvod</b>	<b>4</b>
<b>2. Základní informace o programu Mathematica</b>	<b>5</b>
2.1 Komu je program Mathematica určen?	5
2.2 Co program Mathematica umí?	5
2.3 Jak program Mathematica vznikl?	6
2.4 Uživatelské rozhraní programu Mathematica - notebook	6
2.5 Komunikace s jádrem programu - vstup In[ ] a výstup Out[ ]	7
2.6 Práce s proměnnými	8
2.7 Informace o příkazech programu Mathematica, nápověda a palety	9
2.8 Použití programu Mathematica jako textového editoru	11
<b>3. Řešení aritmetických operací a matematických funkcí</b>	<b>12</b>
3.1 Základní aritmetické operace	12
3.2 Přesnost výpočtů	13
3.3 Základní matematické funkce	14
3.4 Funkce definované uživatelem	19
3.5 Výpočty s komplexními čísly	19
<b>4. Práce s algebraickými výrazy</b>	<b>21</b>
4.1 Zápis a úprava algebraických výrazů	21
4.2 Dosazování proměnných do algebraických výrazů	22
<b>5. Práce s grafikou v rovině</b>	<b>25</b>
5.1 Obecná pravidla pro práci s grafikou	25
5.2 Základní příkazy	25
5.3 Změny stylů kreslení	29
5.4 Úprava zobrazované plochy grafu	32
5.5 Úprava os grafu a oblasti grafu	33
<b>6. Práce s 3D grafikou</b>	<b>39</b>
6.1 Obecná pravidla pro práci s 3D grafikou	39
6.2 Základní příkazy a souřadnicový systém pro 3D grafiku	39
6.3 Úpravy 3D grafů	42
<b>7. Specifické typy grafů využívané v geografii</b>	<b>43</b>
7.1 Trojúhelníkový graf	43
7.2 Věková pyramida	46
<b>8. Práce s funkcemi</b>	<b>48</b>
8.1 Derivace funkcí	48
8.2 Integrál	49
8.3 Limity	51
<b>9. Využití programu Mathematica ve statistice pro geografy</b>	<b>54</b>
9.1 Třídění dat a rozdělení četností	54
9.2 Třídění dat do intervalů	54
9.3 Grafické vyjádření rozdělení četností	55
9.4 Základní statistické charakteristiky : charakteristiky úrovně a polohy	56
9.5 Základní statistické charakteristiky : charakteristiky variability	59
9.6 Základní statistické charakteristiky : charakteristiky šikmosti	60
9.7 Základní statistické charakteristiky : charakteristiky špičatosti	61
9.8 Závislosti mezi náhodnými veličinami	61
<b>10. Export a úprava souborů formátu shp v programu Mathematica</b>	<b>65</b>
<b>11. Využití WolframAlpha v geografii</b>	<b>69</b>
<b>12. Závěr</b>	<b>80</b>
<b>13. Použitá literatura</b>	<b>81</b>

# I Úvod

Matematika je dnes oborem nezbytným pro studium a následnou profesní kariéru v přírodovědných i ekonomických disciplínách a odpovídajících učitelských oborech. V současnosti je třeba si klást otázky: Jakým způsobem je realizována současná výuka matematiky? Je výuka matematiky opravdu totéž jako schopnost počítat nebo provést nějaký výpočet? Jaký je význam matematiky pro profesní kariéru a myšlení mladých lidí, případně pro společnost? V době počítačů, notebooků a tabletů, jejichž výpočetní možnosti mnohonásobně převyšují možnosti kalkulaček, je nevyhnutelné tyto prostředky zahrnout do výuky a vést studenty k jejich smysluplnému používání. Tím však není myšlena pouze aplikace k pohodlnému provedení výpočtů. Díky moderním technologiím vznikl prostor k řešení náročnějších problémů, testování různých přístupů k problémům a diskusi nových kreativních myšlenek. Implementace profesionálního matematického software *Mathematica*, který lze dnes používat kromě počítačů a notebooků také pomocí tabletů nebo prostřednictvím webových rozhraní, do výuky, přispívá dle našeho názoru ke zkvalitnění vyučovacího procesu těchto náročných a často neoblíbených oborů.

Cílem tohoto textu je přiblížit profesionální matematický software *Mathematica* studentům geografie.



## 2 Základní informace o programu Mathematica

*Mathematica* je počítačový program používaný zejména pro řešení technických a matematických úloh, soubory tohoto programu mají příponu \*.nb. Vzhledem k tomu, že matematické výpočty jsou nezbytné prakticky ve všech vědních disciplínách, je program široce využíván prakticky ve všech přírodovědných oborech a v technických vědách, ze společenských věd pak především v ekonomii. Program vyvíjí a prodává firma Wolfram Research se sídlem v Champaign, Illinois v USA, nové verze jsou dostupné pro OS Microsoft Windows, MacOS X a Linux ve třech jazykových mutacích (anglické, čínské a japonské). Kromě standardních verzí je nabízeno velké množství balíčků zaměřených na speciální oblasti, jako například aplikovaná fyzika, finanční matematika, aplikovaná statistika, biomatematika atd..

### 2.1 Komu je program Mathematica určen?

Program vznikl jako „system for doing mathematics by computer“, jak původně zněl jeho podnázev, není tedy primárně určen pouze odborným matematikům, naopak se uživatelsky zaměřuje na „matematické laiky“, kterým se snaží přístupným způsobem zprostředkovat rozsáhlý matematický aparát tak, aby ho mohli plnohodnotně využívat. Protože do kategorie „matematických laiků“ patří i většina geografů, jedná se o program, který může v geovědách najít poměrně vděčné uživatele. To přirozeně neznamená, že pro samotné matematiky je program nezajímavý nebo rutinní: rozsah a kvalita matematických postupů a algoritmů je v něm tak vysoká, že program je vhodným pomocníkem i pro ně.

### 2.2 Co program Mathematica umí?

*Mathematica*, podobně jako další programy určené na ulehčení aplikací matematiky, umožňuje nejen klasické numerické výpočty, ale i výpočty symbolické. Ty se neomezuji na základní matematické operace, ale jsou možné ve všech oblastech od elementární algebry přes matice až k diferenciálnímu a integrálnímu počtu. To umožňuje kvalitativní analýzu zkoumaných vztahů, sledování jejich závislostí na proměnlivých parametrech a další vazby. V případech, kdy není možné najít řešení v obecném tvaru (nelineární rovnice, parciální diferenciální rovnice vyšších řádů, apod.) a je proto nutné kvantitativní vyhodnocení pro konkrétní hodnoty, nabízí program možnost numerického řešení včetně práce s komplexními čísly a možnosti výpočtu s předem stanovenou mírou přesnosti. Silnou stránkou programu je tradičně i propracovaná grafika - od běžných dvourozměrných grafů po grafy prostorové s jednoduchou možností změnit úhel pohledu.

Systém *Mathematica* zahrnuje především nepřeberné množství funkcí a algoritmů z oblasti algebry, geometrie, kombinatoriky, statistiky a regresní analýzy, řešení rovnic a jejich soustav, diferenciálního a integrálního počtu, vizualizace 2D a 3D dat, práce s databázemi a daty (data mining, klastrová analýza, apod.), práce s 2D a 3D obrazy (včetně rozpoznávání obrazu), funkce pro kombinatoriku, analýzu textu, funkce a algoritmy z teorie čísel, prostředky pro výpočty v ekonomii, apod. *Mathematica* umožňuje import a export dat, obrázků, videa, zvuku, GIS dokumentů a biomedicínských programů. Pokročilejší uživatelé na programu oceňují i jeho výborný programovací jazyk podobný C++.

Pro geovědy je důležitou funkcí programu možnost importovat data z neustále rozšiřované databáze matematických, vědeckých a socioekonomických informací, která je pro uživatele programu dostupná online prostřednictvím služby WolframAlpha. Některá z dat dostupných na WolframAlpha jsou aktualizována v reálném čase (např. data o počasí). Databáze shromažďuje údaje ze všech oborů, k dispozici jsou mimo jiné i demografická data, geopolitická údaje, hospodářská data, apod. Wolfram Alpha funguje i samostatně jako „odpovídající stroj“ (tj. na rozdíl od vyhledavačů, které poskytují seznam stránek, na kterých by mohla být odpověď

obsažena, nabízí přímo odpověď nebo požadovaný údaj). U dotazů matematických WolframAlpha nabízí i postup vedoucí k výsledku, výhodou pro uživatele je také možnost zadávat otázky přirozeným jazykem (přesněji anglicky, např.: „Where was Barack Obama born?“).

## 2.3 Jak program Mathematica vznikl?

Program *Mathematica* byl vytvořen Stephenem Wolframem (\* 29. 8. 1959 v Londýně, oba rodiče byli uprchlíci z hitlerovského Německa) v roce 1988 na základě staršího programu SMP vyvíjeného od roku 1979. Pro jeho programování byl vytvořen speciální programovací jazyk označovaný podle svého tvůrce jako „Wolfram Language“. V následujících letech byly postupně uvolňovány na trh nové verze vyvíjené týmem matematiků a programátorů soustředěných kolem Wolframa, resp. kolem jeho firmy Wolfram Research. Současná verze, *Mathematica* 10.0.1, která je v prodeji od 17. září 2014, je už 2.3. variantou programu.

## 2.4 Uživatelské rozhraní programu Mathematica - notebook

*Mathematica* je rozdělena do dvou částí – jádra a front endu. Jádro interpretuje výrazy a vrací výsledky. Front end poskytuje grafické uživatelské rozhraní, ve kterém výsledky vhodně zobrazuje.

Front end programu *Mathematica* vytvořil spoluzakladatel firmy Wolfram Research Theodore Gray, který mu dal graficky velmi střízlivou podobu zápisníku (čeští uživatelé mu říkají notebook). Notebook umožňuje komunikaci s jádrem programu formou vstupu a výstupu, textové formátování včetně zápisu složitějších matematických výrazů, formátování grafických výstupů, tabulek a zvuku. Obsah a formátování může být generováno algoritmicky nebo interaktivně (případně kombinací obou přístupů). Notebook je možné uložit, kopírovat, dodatečně upravovat, vytisknout apod., nemusí tedy sloužit výlučně jen ke komunikaci s jádrem programu, ale také k prezentaci a formátování výsledků.

Zápisník sice vypadá jako běžný textový editor, má ale některá neobvyklá specifika. Předně je členěn na buňky, jejichž začátek a konec ukazují hranaté závorky na pravé straně notebooku. Počáteční poloha kurzoru na začátku buňky není svislá, ale vodorovná. Buňky mohou být několika typů, základní jsou:

- a) vstupní – input (dotaz pro jádro programu – o nich více v následující kapitole),
- b) výstupní – output (odpověď jádra programu),
- c) textové (neslouží ke komunikaci s jádrem programu, ale umožňují např. psát poznámky k postupu, vysvětlivky, apod.).

Vstupní buňku uzavřeme příkazem na zpracování do jádra programu (viz následující kapitola), textová buňka se uzavírá nastavením vodorovného kurzoru myši pod text a stlačením levého tlačítka myši (přes celý notebook se objeví vodorovná čára a kurzor se nastaví na počátek nové buňky). Stejně se postupuje, pokud potřebujeme vložit novou buňku mezi dvě už existující buňky.

S buňkami je možné pracovat obdobně jako v běžných textových editorech s odstavci: je možné je přesouvat, kopírovat, mazat, formátovat jejich písmo i pozadí, apod. Pro tyto operace je však nutné nejdříve buňku označit kliknutím myši na hranatou závorku na pravé straně notebooku (kurzor se předtím změní na šipku doleva), označená buňka pak má při pravém okraji notebooku tlustou plnou svislou čáru. Výběr zrušíme kliknutím myši kamkoliv mimo pravý okraj buňky.

Přejetím myši přes více po sobě následujících závorek označujících buňky je možné vybrat skupinu buněk. Takto označené buňky je pak možné např. spojit do jedné buňky, seskupit je, animovat (pokud jsou grafické), apod. Nabídku možných operací dostaneme kliknutím pravého tlačítka myši na pravý okraj buňky.

## 2.5 Komunikace s jádrem programu - vstup In[ ] a výstup Out[ ]

Komunikace mezi jádrem programu a notebookem se děje pomocí vstupu (náš dotaz) a výstupu (odpověď programu). Vstup zadáme v notebooku tak, že napíšeme výraz, který chceme zpracovat, a stiskneme Shift+Enter (nebo ikonou *Mathematica* z grafického menu, nebo Ins, nebo 5 na numerické klávesnici). Naše vstupy označuje program postupně jako In[1]:=, In[2]:=, In[3]:=, In[4]:= a okamžitě po zadání na ně odpovídá, odpověď na n-tý vstup označuje jako Out[n]:=.

Příklady:

In[1]:= **1 + 2**

Out[1]= 3

In[2]:= **20 \* 5 - 4**

Out[2]= 96

Vstup může být i delší než jeden řádek, aby byl ale jádrem programu „pochopen“, musí každý z řádků končit symbolem, který signalizuje, že příkaz pokračuje na dalším řádku (např. matematický operátor, šipka, apod.).

Příklad:

In[3]:= **(14 - 4) \* (5 + 5)**

Out[3]= 100

Jeden vstup může obsahovat i více příkazů zároveň, jednotlivé příkazy se oddělují středníkem. Když pak takovou skupinu příkazů odešleme na zpracování, vykonají se všechny, ve výstupu Out[n]:= se ale objeví jen odpověď na příkaz poslední. Proto se doporučuje psát do jednoho vstupu příkazy, které spolu nějak souvisí, nebo na sebe navazují. Nedoporučuje se spojovat do jednoho vstupu příkazy na export dat a na jejich zpracování, rovněž je nevhodné kombinovat takto příkazy určené k jednorázovému a k opakovanému užití.

Příklad:

In[4]:= **a = 3; b = 3; a + b**

Out[4]= 6

Pokud se rozhodneme dodatečně příkazy zapsané do jednoho vstupu rozdělit do dvou buněk, vytvoříme pomocí klávesy Enter prázdný řádek a následně Shift + Ctrl + D. V příkazech můžeme použít výsledky předchozích výstupů (ať už jde o čísla, výrazy, grafiku, nebo cokoliv jiného). Na poslední výsledek odkazuje symbol %, na předposlední %, výsledek z Out[n] se pak může označit %n nebo přímo Out[n].

Příklad:

In[5]:= **2 \* %**

Out[5]= 12

In[6]:= **% / 2**

Out[6]= 6

In[7]:= **%% - 11**

Out[7]= 1

## 2.6 Práce s proměnnými

Jak je patrné z předchozího příkladu, odvolávání se na výsledky n-tého výstupu je poměrně těžkopádné a nepřehledné. V praxi se prakticky nepoužívá, výsledek příkazu, který máme v úmyslu dále zpracovávat nebo opakovaně používat, je vhodnější přiřadit nějaké proměnné (např. výsledek výpočtu označíme písmenem  $q$  a vždy, když budeme v dalších výpočtech tuto hodnotu používat, do výrazu doplníme pouze  $q$ ).

Proměnná může být označena libovolně dlouhým označením, je třeba ale respektovat 2 pravidla:

- první znak označení musí být písmeno, na dalších místech mohou být písmena a číslice,
- označení proměnné se nesmí shodovat s některým z příkazů, které používá počítačový program *Mathematica* (např. `Sin` je označení pro funkci sinus, žádná proměnná se tedy „`Sin`“ jmenovat nesmí).

Protože všechny příkazy v programu *Mathematica* začínají velkým počátečním písmenem, doporučuje se, aby jména proměnných začínala písmenem malým. Pak nemůže dojít k „nedorozumění“, protože systém důsledně odlišuje malá a velká písmena. Tento fakt je třeba mít vždy na paměti, protože např. na proměnnou označenou jako `a11` se nemůžeme odvolávat výrazem „`A11`“. Přiřazení konkrétní hodnoty proměnné se provádí pomocí operátoru `=`. Stejnou hodnotu můžeme přiřadit i více proměnným současně.

Příklad:

```
In[8]:= a = 3; b = c = 2
```

```
Out[8]= 2
```

```
In[9]:= a - b + c
```

```
Out[9]= 3
```

Jednou přiřazenou hodnotu proměnné si jádro programu „pamatuje“, dokud ji nenahradíme jinou hodnotou nebo dokud proměnnou nezrušíme. Fakt, že jsou hodnoty proměnných „evidovány“ přímo jádrem programu, je důležitý hlavně v případech, kdy pracujeme s více notebooky současně, nebo když ukončíme práci s jedním a bezprostředně pokračujeme v práci s dalším notebookem. Přepsání proměnné na novou hodnotu může vypadat např. takto:

```
In[10]:= a = 3
```

```
Out[10]= 3
```

```
In[11]:= 4 * a
```

```
Out[11]= 12
```

```
In[12]:= a = 12
```

```
Out[12]= 12
```

```
In[13]:= a - 11
```

```
Out[13]= 1
```

Přirozeně můžeme proměnné přiřadit i nějaký výsledek matematické operace, ve které je proměnná použita. Pokud bychom proměnnou `a` z předchozího příkladu (v tuto chvíli má hodnotu 12) chtěli zdvojnásobit, stačí použít příkaz (upozorňujeme znovu, že `=` je operátor přiřazení hodnoty):

```
In[14]:= a = 2 * a
```

```
Out[14]= 24
```

Hodnotu proměnné lze vymazat příkazem `Clear[x]`, nebo `x=.`. Vymazání hodnoty proměnné po jejím posledním použití nás může ochránit před omyly nebo nedorozuměními při dalším použití programu. Vymazat lze i několik proměnných současně, příslušný příkaz má podobu `Clear[x, y, z]`, případně `x=y=z=.`

## 2.7 Informace o příkazech programu Mathematica, nápověda a palety

Podat v tomto textu úplný výčet příkazů programu *Mathematica* by bylo zbytečné. Postupně se seznámíme s těmi základními, specializované příkazy pak může zájemce poznat v podstatě dvěma způsoby:

- v nápovědě programu (záložka Help – Function Navigator),
- přímo dotazem v notebooku.

Dotaz v notebooku má syntaxi `?NazevPrikazu` pro základní informace o příkazu, `??NazevPrikazu` pro podrobnější informace a `?Nazev*` pro vyhledání všech příkazů začínajících písmeny „Nazev“ (znak `*` pro vynechaná písmena může být i uvnitř nebo na začátku slova). Vstup s dotazem se nekombinuje s jinými příkazy, proto musí být otazník v dotazu vždy na začátku buňky. Pokud chceme např. zjistit, co přesně znamená příkaz `Sin` a jaká je jeho přesná syntaxe, můžeme postupovat takto:

In[15]:= `? Sin`

`Sin[z]` gives the sine of z. >>

In[16]:= `?? Sin`

`Sin[z]` gives the sine of z. >>

`Attributes[Sin] = {Listable, NumericFunction, Protected}`

In[17]:= `? Sin*`

▼ System`

<code>Sin</code>	<code>SingleLetterItalics</code>	<code>SingularValuePlot</code>	<code>SinIntegral</code>
<code>Sinc</code>	<code>SingleLetterStyle</code>	<code>SingularValues</code>	
<code>SinghMaddalaDistribution</code>	<code>SingularValueDecomposition</code>	<code>Sinh</code>	
<code>SingleEvaluation</code>	<code>SingularValueList</code>	<code>SinhIntegral</code>	

Kliknutím na symbol `>>` v odpovědi na dotaz se pak dostaneme na podrobné informace o příkazu v nápovědě, které obsahují také základní i pokročilé příklady užití. V případě funkce `Sin` jde např. o tyto doplňkové informace:

Mathematical function, suitable for both symbolic and numerical manipulation.

The argument of `Sin` is assumed to be in radians. (Multiply by `Degree` to convert from degrees.)

`Sin` is automatically evaluated when its argument is a simple rational multiple of  $\pi$ ; for more complicated rational multiples, `FunctionExpand` can sometimes be used.

For certain special arguments, `Sin` automatically evaluates to exact values.

`Sin` can be evaluated to arbitrary numerical precision.

`Sin` automatically threads over lists.

Základní příklady použití (v nápovědě je 51 příkladů, zde ukázány jen 4 nejzákladnější):

The argument is given in radians:

In[18]:= **Sin**[Pi / 3]

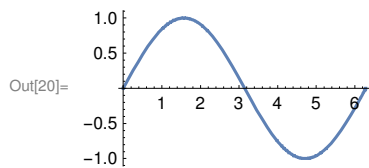
Out[18]=  $\frac{\sqrt{3}}{2}$

Use Degree to specify an argument in degrees:

In[19]:= **Sin**[60 Degree]

Out[19]=  $\frac{\sqrt{3}}{2}$

In[20]:= **Plot**[Sin[x], {x, 0, 2 Pi}]



In[21]:= **Series**[Sin[x], {x, 0, 10}]

Out[21]=  $x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040} + \frac{x^9}{362880} + O[x]^{11}$

Zejména pro začátečníky je výhodné používat při práci s notebookem palety. Palety se nacházejí v horním řádku programu pod názvem Palettes. Při práci v notebooku jsou velmi užitečné zejména paleta Basic Math Assistant, která obsahuje další podkategorie. Kategorie Calculator obsahuje syntaxi příkazů pro aritmetiku, algebru, funkce, diferenciální počet, apod., např.:

$$\int_{\text{lower}}^{\text{upper}} \text{expr} \, d\text{var}$$

Basic Commands obsahuje matematické konstanty, elementární funkce, nabízí jednotlivé vlastnosti tabulek, 2D a 3D grafů apod., např.:

**Sort** [list, Greater]

**Plot3D** [function, {var<sub>1</sub>, min, max}, {var<sub>2</sub>, min, max}]

Další velmi užitečná paleta je Writing Assistant. I ta má několik podkategorií, z nichž ta nazvaná Writting and Formatting umožňuje práci s textem (tučně, kurzíva, velikost textu ap.), dále umožňuje úpravy jednotlivých buněk (pozadí buněk, orámování buněk), zarovnání textu, lze také nastavit hranice jednotlivých stránek v prezentaci (Start Slide, End Slide), atd. Podkategorie Typesetting pak umožňuje zadat písmena řecké abecedy, speciální symboly, tvary či operátory.

## 2.8 Použití programu Mathematica jako textového editoru

Jak jsme již uvedli, s buňkami notebooku se pracuje podobně jako s textovým editorem. Program je proto možné použít i k editování nebo prezentování jednodušších textů, zejména v případech, kdy chceme prezentovat výsledky práce v programu *Mathematica*. Text v notebooku je možné uložit i ve formátech přístupných běžným textovým editorům (TeX, XML, RTF pro Word, PDF, ale také jen jako čistý text TXT, nebo ve formátu HTML. V této souvislosti je nicméně třeba poznamenat, že se *Mathematica* nehodí k editování dlouhých textů a že jsou značně omezeny i možnosti text formátovat, nelze proto očekávat výsledky srovnatelné s programy speciálně vyvinutými pro editaci textů. Protože není program vyvíjen v české jazykové verzi, mohou při převodu českých textů nastat i problémy se zobrazováním některých znaků (např. u některých starších verzí programu při převodu do \*.pdf) nebo s typografií (zejména v případech, kdy je v češtině odlišná od anglické), na rozdíl od angličtiny také není možné používat automatické kontroly pravopisu a gramatiky.

Možnost strukturovat text jako slideshow ocení zejména ti, kteří chtějí výstupy z programu *Mathematica* veřejně prezentovat.

## 3 Řešení aritmetických operací a matematických funkcí

### 3.1 Základní aritmetické operace

Program *Mathematica* lze přirozeně používat i pro výpočet aritmetických operací, tj. podobně jako kalkulačku. Syntaxe základních aritmetických operací je následující:

<b>2 . 45</b>	desetinná čísla	2, 45
<b>a + b</b>	sčítání	<b>a + b</b>
<b>a - b</b>	odčítání	<b>a - b</b>
<b>a * b</b> , nebo <b>a b</b> , nebo <b>3 c</b>	násobení	<b>ab</b> , <b>3 c</b>
<b>a / b</b>	dělení	$\frac{a}{b}$
<b>a ^ b</b>	mocnina	$a^b$
<b>()</b>	kulaté závorky	<b>()</b>

Desetinná čísla je třeba psát s anglickou typografií, tj. místo desetinné čárky používat tečku. Všimněte si, že pro násobení se používá operátor `*` nebo mezera. Mezera v tomto případě nelze vynechat, protože program by zadání nerozuměl: výraz `a b` *Mathematica* interpretuje jako „a krát b“, ale výraz `ab` je pro ni proměnná s označením `ab`. Protože však jméno každé proměnné začíná malým počátečním písmenem, je možné při násobení čísla a proměnné (`2*c`, `2 c`) mezera úplně vynechat (`2c`). Obdobný je postup, pokud násobíme číslem výraz v závorce.

Při aritmetických výpočtech postupuje program zleva doprava, nejvyšší prioritu má umocňování, pak násobení s dělením a jako poslední sčítání s odčítáním. Prioritu operátorů lze měnit jen kulatými závorkami, ostatní typy závorek se v *Mathematice* používají k jiným účelům.

Příklady:

In[22]:= **5 + 2 × 5**

Out[22]= 15

In[23]:= **a = 7; 3 a**

Out[23]= 21

In[24]:= **(2 - 3 (4 - 3))**

Out[24]= -1

In[25]:= **-1 (2 - 3 (4 - 3))**

Out[25]= 1

In[26]:= **1 × 3 / 4**

Out[26]=  $\frac{3}{4}$

In[27]:= **15.67 / 10**

Out[27]= 1.567



```
In[28]:= 1.567 - 1.567
```

```
Out[28]= 0.
```

Pozor na odlišnosti v zápisu mocnin  $2^{3^2}$  a  $(2^3)^2$ :

```
In[29]:= 2^3^2
```

```
Out[29]= 512
```

```
In[30]:= (2^3)^2
```

```
Out[30]= 64
```

## 3.2 Přesnost výpočtů

Oproti běžným kalkulačkám je program *Mathematica* schopen spočítat přesně i velmi malá nebo naopak velmi velká čísla, která by se na displej běžné kalkulačky jednoduše nevešla. Např. výpočet  $21^{21}$ :

```
In[31]:= 21^21
```

```
Out[31]= 5 842 587 018 385 982 521 381 124 421
```

Schopnost vypočítat zcela přesné výsledky je v matematice velmi potřebná, ve statistických výpočtech v geografii je ale obvyklejší, že chceme zapsat výsledek v jednoduché a přehledné formě, např. 10,3 mil. (tedy  $1,03 \cdot 10^7$ ) místo „přesných“ 10 456 789, nebo 1,15 mm (tj.  $1,15 \cdot 10^{-3}$ ) místo přesných 0,001 149 896 780 m. V těchto případech nám tedy jde o přibližné hodnoty výsledku. K získání přibližné numerické hodnoty výrazu můžeme použít příkazy `N[výraz]` nebo `výraz/N` (výsledek má pak 6 platných číslic); přibližnou numerickou hodnotu vypočtenou na `k` platných číslic získáme použitím příkazu `N[výraz,k]`. Pokud je aplikujeme na předchozí příklad:

```
In[32]:= N[21^21]
```

```
Out[32]= 5.84259 × 1027
```

```
In[33]:= 21^21 // N
```

```
Out[33]= 5.84259 × 1027
```

```
In[34]:= N[21^21, 4]
```

```
Out[34]= 5.843 × 1027
```

```
In[35]:= N[21^21, 2]
```

```
Out[35]= 5.8 × 1027
```

Stejný efekt, jako požadavek na přibližnou numerickou hodnotu pomocí příkazu `N[výraz]` má, když je ve výrazu alespoň jedno číslo zapsané s desetinnou tečkou. Systém pak požadavek vyhodnotí jako žádost o přibližnou hodnotu výrazu. Z celého čísla přitom můžeme „udělat“ reálné (desetinné) pouhým přidáním desetinné tečky, není ani nutno za ni doplňovat nulu. V následujících příkladech si všimněte rozdílu: žádost o přesný výsledek vypadá takto (všechna čísla jsou celá nebo zapsaná v symbolickém tvaru):

```
In[36]:= 21^21
```

```
Out[36]= 5 842 587 018 385 982 521 381 124 421
```

In[37]:=  $5 * (\sqrt{3} - \sqrt{12})$

Out[37]:=  $-5 \sqrt{3}$

Z posledního příkladu můžeme udělat žádost o přibližný výsledek (ve tvaru desetinného čísla) jen tím, že např. číslo 3 zapíšeme jako reálné (3.):

In[38]:=  $5 * (\sqrt{3.} - \sqrt{12})$

Out[38]:=  $-8.66025$

V případě, že nepotřebujeme přesný výsledek, doporučuje se používat přibližné výpočty, protože jsou podstatně rychlejší. Že jde ale skutečně “jen” o přibližné výsledky, si můžeme ukázat na jednoduchém příkladu, u kterého je na první pohled jasné, že přesný výsledek je nulový:

In[39]:=  $1.567 - 15.67 / 10$

Out[39]:=  $-2.22045 \times 10^{-16}$

Evidentně se výsledek od „přesné“ nuly liší, rozdíl je ale zanedbatelný (pokud by např. byly údaje ve výpočtu v kilometrech, míra „nepřesnosti“ předchozího přibližného výpočtu by byla asi 22 pikometrů, což je řádově velikost několika atomů. Kdyby chtěl nějaký „šřoural“ získat přeci jen zcela přesný výsledek, musel by zapsat desetinná čísla jako zlomky:

In[40]:=  $1567 / 1000 - (1567 / 100) / 10$

Out[40]= 0

### 3.3 Základní matematické funkce

Jak jsme již uvedli, jsou do programu *Mathematica* včleněny všechny běžně používané matematické funkce. Názvy funkcí mají vždy velké počáteční písmeno, jejich proměnné a různé parametry se uvádějí vždy v hranatých závorkách (nelze je zaměnit jiným typem závorek). Názvy funkcí musí být vypisovány zcela přesně, pokud uděláme chybu v syntaxi příkazu, jádro programu nevrátí výsledek, ale upozornění na chybu. Protože některé názvy funkcí jsou poměrně dlouhé, je možné zápis urychlit např. volbou příkazu Complete selection v záložce Edit (nebo klávesovou zkratkou Ctrl + K). Program pak nabídne na výběr všechny funkce, jejichž název začíná zadaným písmenem nebo skupinou písmen. V následující tabulce uvedeme jen ty nejzákladnější a nejčastěji používané funkce a příkazy:

<b>Sqrt [x]</b>	odmocnina, $\sqrt{x}$
<b>Exp [x]</b>	exponenciální funkce o základu e, $e^x$
<b>Log [x]</b>	přirozený logaritmus, $\ln x$
<b>Log [a, x]</b>	logaritmus o základu a, $\log_a x$
<b>Sin [x]</b>	sinus (hodnota x v radiánech), $\sin (x)$
<b>Cos [x]</b>	kosinus (hodnota x v radiánech), $\cos (x)$
<b>Tan [x]</b>	tangens (hodnota x v radiánech), $\text{tg} (x)$
<b>ArcSin [x], ArcCos [x], ArcTan [x]</b>	inverzní trigonometrické funkce, arcsin (x), arccos (x), arctg (x)
<b>n !</b>	n – faktoriál
<b>Abs [x]</b>	absolutní hodnota, $ x $
<b>Round [x]</b>	zaokrouhlení na celá čísla (tj. nejbližší celé číslo)

<b>Ceiling</b> [ <i>x</i> ]	zaokrouhlení na celá čísla nahoru (tj. nejmenší celé číslo, které není menší než <i>x</i> )
<b>Floor</b> [ <i>x</i> ]	zaokrouhlení na celá čísla dolů (tj. největší celé číslo, které není větší než <i>x</i> )
<b>Max</b> [ <i>x</i> , <i>y</i> , ...]	maximum z čísel <i>x</i> , <i>y</i> , ...
<b>Min</b> [ <i>x</i> , <i>y</i> , ...]	minimum z čísel <i>x</i> , <i>y</i> , ...
<b>Sum</b> [ <i>x</i> , <i>y</i> , ...]	součet čísel <i>x</i> , <i>y</i> , ...
<b>Sum</b> [ <i>f</i> ( <i>x<sub>i</sub></i> ), { <i>i</i> , <i>c</i> }]	součet $\sum_{i=1}^c f(x_i)$
<b>Sum</b> [ <i>f</i> ( <i>x<sub>i</sub></i> ), { <i>i</i> , <i>a</i> , <i>b</i> }]	součet $\sum_{i=a}^b f(x_i)$
<b>Sum</b> [ <i>f</i> ( <i>x<sub>i</sub></i> ), { <i>i</i> , <i>a</i> , <i>b</i> , <i>h</i> }]	součet <i>f</i> ( <i>x<sub>i</sub></i> ) pro <i>i</i> = <i>a</i> , <i>a</i> + <i>h</i> , <i>a</i> + 2 <i>h</i> , <i>a</i> + 3 <i>h</i> , ..., <i>b</i>
<b>Sum</b> [ <i>f</i> ( <i>x<sub>ij</sub></i> ), { <i>i</i> , <i>a</i> , <i>b</i> }, { <i>j</i> , <i>c</i> , <i>d</i> }]	součet $\sum_{i=a}^b \sum_{j=c}^d f(x_{ij})$
<b>Product</b> [ <i>f</i> ( <i>x<sub>i</sub></i> ), { <i>i</i> , <i>a</i> , <i>b</i> }]	součin $\prod_{i=a}^b f(x_i)$
<b>NSum</b> [...], <b>NProduct</b> [...]	numerická aproximace součtu, součinu

Před samotnými příklady použití základních matematických funkcí je nutné také upozornit na některé častěji používané konstanty. Ty si můžeme přirozeně sami nadefinovat (jako proměnné), program *Mathematica* má ale ty častěji používané už předdefinované (na rozdíl od proměnných jejich název začíná velkým písmenem, podobně jako u příkazů):

<b>Pi</b>	Ludolfovo číslo, $\pi = 3, 14 159 \dots$
<b>E</b>	Eulerovo číslo; základ přirozeného logaritmu, $e = 2, 71 828 \dots$
<b>Degree</b>	konstanta pro převod radiánů na stupně, $\frac{\pi}{180}$
<b>I</b>	imaginární jednotka komplexního čísla, $i = \sqrt{-1}$ , $i^2 = -1$
<b>Infinity</b>	nekonečno, $\infty$

## Výpočet druhé odmocniny:

In[41]:= **Sqrt**[9]

Out[41]= 3

Pokud nám zápis dotazu v textové podobě z nějakého důvodu nevyhovuje, můžeme přirozeně použít i zápis symbolický (v praxi se ale tento způsob zápisu nepoužívá, protože je pomalejší, potřebné operátory musíme totiž vyhledávat v záložce Palettes - Basic Math Assistant):

In[42]:=  $\sqrt{9}$

Out[42]= 3

Pozor na rozdíl mezi přesnými a přibližnými výpočty. Pokud zadáme výpočet jako přesný a výsledek není celé číslo, vrátí *Mathematica* výsledek v symbolickém tvaru:

In[43]:= **Sqrt**[5]

Out[43]=  $\sqrt{5}$

Pro výsledek v podobě desetinného čísla musíme systému "sdělit", že stojíme o přibližný výsledek, tj. použijeme některou z možností:

```
In[44]:= N[Sqrt[5]]
```

```
Out[44]= 2.23607
```

```
In[45]:= Sqrt[5] // N
```

```
Out[45]= 2.23607
```

```
In[46]:= Sqrt[5.]
```

```
Out[46]= 2.23607
```

## Výpočty trigonometrických funkcí:

```
In[47]:= Sin[Pi / 4]
```

```
Out[47]=  $\frac{1}{\sqrt{2}}$ 
```

Předchozí výpočet je přesný, pro přibližný výpočet můžeme např. použít zápisu:

```
In[48]:= Sin[Pi / 4.]
```

```
Out[48]= 0.707107
```

Pokud nezadááme argument v radiánech, ale ve stupních, nesmíme zapomenout na jeho převod na radiány. Nejjednodušší je v tomto případě prosté vynásobení argumentu předdefinovanou konstantou Degree (tj.  $\frac{\pi}{180}$ ).

Upozorňujeme, že „Degree“ není pro program *Mathematica* označení jednotky, ale reálné číslo, proto je nutné dodržet syntaxi pro násobení (připomínáme, že pro násobení lze použít znak \* nebo mezeru):

```
In[49]:= Tan[135 * Degree]
```

```
Out[49]= -1
```

resp. v podobě „Tan[135 Degree]“:

```
In[50]:= Tan[135 Degree]
```

```
Out[50]= -1
```

Příklad chybné syntaxe (bez pro program srozumitelného označení násobení konstantou Degree):

```
In[51]:= a = 45; Tan [aDegree]
```

```
Out[51]= Tan [aDegree]
```

```
In[52]:= Tan [a Degree]
```

```
Out[52]= 1
```

## Výpočty exponenciálních funkcí a logaritmů:

Hodnotu  $e^2$  můžeme vypočítat dvěma způsoby:

```
In[53]:= Exp[2.]
```

```
Out[53]= 7.38906
```

```
In[54]:= E ^ 2 .
Out[54]= 7.38906
```

Výpočty logaritmů by neměly být spojeny s žádnými problémy. Ukážeme si postupně výpočty  $\ln 10$ ,  $\log_8 64$  a  $\log_2 512$ .

```
In[55]:= Log[10.]
Out[55]= 2.30259
```

```
In[56]:= Log[8, 64.]
Out[56]= 2.
```

```
In[57]:= Log[2, 512]
Out[57]= 9
```

Všimněte si, že druhý výpočet jsme zadali jako přibližný (proto je výsledek 2, 0), následující výpočet je pak přesný (výsledek 8). V případě  $\ln 10$  by program poskytl přesný výsledek v symbolickém tvaru:

```
In[58]:= Log[10]
Out[58]= Log[10]
```

---

## Zaokrouhlování:

Rozdíly mezi funkcemi `Round[x]`, `Floor[x]` a `Ceiling[x]` si ukážeme na jednoduchém příkladu. Pro kladné číslo 12,678 “dopadne” zaokrouhlování takto:

```
In[59]:= Round[12.678]
Out[59]= 13
```

```
In[60]:= Floor[12.678]
Out[60]= 12
```

```
In[61]:= Ceiling[12.678]
Out[61]= 13
```

Pro srovnání: záporné číslo -12,678 (v zápisu si pomůžeme tím, že tuto hodnotu přiřadíme proměnné označené `d`)

```
In[62]:= d = -12.678
Out[62]= -12.678
```

```
In[63]:= Round[d]
Out[63]= -13
```

```
In[64]:= Floor[d]
Out[64]= -13
```

```
In[65]:= Ceiling[d]
Out[65]= -12
```

## Součty a součiny řad:

Součet prvních 8 členů posloupnosti  $2^i - i$ , tj.  $1+2+5+12+\dots$ , resp.  $\sum_{i=1}^8 (2^i - i)$ :

```
In[66]:= Sum[2^i - i, {i, 8}]
```

```
Out[66]= 474
```

Pro výpočet aritmetického průměru čísel 10, 10.5, 11, 11.5 ... 29.5, 30 můžeme využít součtu řady “po krocích” odlišných od jedničky:

```
In[67]:= Sum[10 + i, {i, 10, 30, 0.5}] / 41
```

```
Out[67]= 30.
```

Při výpočtu nekonečné řady postupujeme obdobně, jen horní mezí nebude konkrétní číslo, ale konstanta označující nekonečno (Infinity). Např. pro  $\sum_{i=1}^{\infty} \frac{1}{2^i}$ :

```
In[68]:= Sum[1 / 2^i, {i, Infinity}]
```

```
Out[68]= 1
```

Výpočet “součtu součtů”  $\sum_{i=1}^{10} \sum_{j=1}^{10} (i + i^j)$  by pak mohl být proveden např. takto:

```
In[69]:= Sum[i + i^j, {i, 1, 10}, {j, 1, 10}]
```

```
Out[69]= 16 676 687 246
```

Nebo jednodušší syntaxí (jak  $i_{\min}$ , tak  $j_{\min}$  je 1):

```
In[70]:= Sum[i + i^j, {i, 10}, {j, 10}]
```

```
Out[70]= 16 676 687 246
```

Názornější je v tomto případě přirozeně uvedení výsledku v přibližném (numerickém) tvaru:

```
In[71]:= N[%]
```

```
Out[71]= 1.66767 × 1010
```

Ten bychom ostatně obdrželi rovnou, pokud si o něj řekneme zadáním funkce NSum:

```
In[72]:= NSum[i + i^j, {i, 10}, {j, 10}]
```

```
Out[72]= 1.66767 × 1010
```

Konečný součin  $\prod_{i=1}^{10} \frac{i - \frac{1}{2}}{i + 1}$  vypočteme takto:

```
In[73]:= Product[ $\frac{i - \frac{1}{2}}{i + 1}$ , {i, 1, 10}]
```

```
Out[73]=  $\frac{4199}{262144}$ 
```

Nekonečný součin  $\prod_{i=1}^{\infty} \frac{i - \frac{1}{2}}{i + 1}$  pak takto:

```
In[74]:= Product[ $\frac{i - \frac{1}{2}}{i + 1}$ , {i, 1, Infinity}]
```

```
Out[74]= 0
```

### 3.4 Funkce definované uživatelem

V programu *Mathematica* je možné velmi snadno definovat uživatelské funkce a tzv. procedury (procedura umožňuje opakování stejného příkazu nebo posloupnosti příkazů na různých místech notebooku). Funkce a procedury se definují stejným způsobem:

<b>f[x_] := výraz</b>	definice funkce f jedné proměnné
<b>f[x_, y_, ...] := výraz</b>	definice funkce f více proměnných
<b>? f</b>	zobrazení definice funkce f
<b>Clear[f]</b>	zrušení definice funkce

Názvy uživatelských funkcí se řídí stejnými pravidly jako názvy proměnných: tj. mohou být označeny libovolně dlouhým označením, je třeba ale respektovat 3 pravidla:

- první znak označení musí být písmeno, na dalších místech mohou být písmena a číslice,
- označení uživatelské funkce se nesmí shodovat s některým z příkazů, které používá počítačový program *Mathematica* (např. *Thickness* je příkaz určující tloušťku čar, žádná uživatelská funkce se tedy „*Thickness*” jmenovat nesmí),
- označení funkce se nesmí shodovat s názvem nějaké dříve nadefinované proměnné.

Protože všechny příkazy v programu *Mathematica* začínají velkým počátečním písmenem, doporučuje se, aby jména uživatelských funkcí začínala písmenem malým.

Vyčíslení uživatelské funkce má stejnou syntaxi jako u funkcí předdefinovaných: napíše se její název a do hranatých závorek se umístí místo formálních proměnných konkrétní hodnoty.

Příklady:

```
In[75]:= funkce007[x_] := x^2 - 2
```

```
In[76]:= funkce007[4]
```

```
Out[76]= 14
```

Při definování uživatelských funkcí je třeba důsledně dbát na syntaxi - nesprávně definovanou funkci *Mathematica* “pochopí” chybně a odkazy na ni pak neprovádí:

```
In[77]:= funkce_spatna[x] := x^2 - 2
```

```
In[78]:= funkce_spatna[3]
```

```
Out[78]= funkce_spatna[3]
```

### 3.5 Výpočty s komplexními čísly

Ve výpočtech je přirozeně možné pracovat nejen s reálnými, ale i s komplexními čísly. V programu *Mathematica* se zapisují ve formě **a+bI**.

<b>a + b I</b>	zápis komplexního čísla $z = a + bi$
<b>Re [z]</b>	reálná část komplexního čísla $z$
<b>Im [z]</b>	imaginární část komplexního čísla $z$
<b>Conjugate [z]</b>	$\bar{z}$ , číslo komplexně sdružené k číslu $z$ (pokud je $z = a + bi$ , pak $\bar{z} = a - bi$ )
<b>Abs [z]</b>	absolutní hodnota komplexního čísla $z$ , tj. $ z  = \sqrt{a^2 + b^2}$
<b>Arg [z]</b>	argument $\varphi$ goniometrického tvaru komplexního čísla, tj. $z =  z  (\cos \varphi + i \sin \varphi)$

Příklady použití:

In[79]:= **z = 4 + I 3**

Out[79]=  $4 + 3 i$

In[80]:= **Re [z]**

Out[80]= 4

In[81]:= **Im [z]**

Out[81]= 3

In[82]:= **Conjugate [z]**

Out[82]=  $4 - 3 i$

In[83]:= **Abs [z]**

Out[83]= 5

In[84]:= **Arg [z]**

Out[84]=  $\text{ArcTan}\left[\frac{3}{4}\right]$

In[85]:= **N [Arg [z]]**

Out[85]= 0.643501



## 4 Práce s algebraickými výrazy

Algebraický výraz je každý matematický zápis, který je tvořen z konstant a proměnných, mezi nimiž jsou pomocí algebraických operací (např. sčítání, násobení) a závorek vytvořeny smysluplné vztahy. S algebraickými výrazy se v matematickém textu setkáváme velmi často, jsou totiž přehledné a srozumitelnější než zdlouhavý slovní popis. Např. algebraický výraz  $\frac{5(x+y)}{\sqrt{x-y}}$  je rozhodně pro čtenáře pochopitelnější, než slovní popsání stejné situace, které by mělo pravděpodobně tuto podobu: “podíl pětinasobku součtu dvou reálných čísel a druhé odmocniny z jejich rozdílu”.

System *Mathematica* umožňuje symbolické výpočty, tj. může zpracovávat algebraické výrazy (např.  $5+x^2$ ) stejně jako výrazy číselné (např.  $5+4^2$ ). Nejčastěji uživatelé programu chtějí algebraické výrazy zjednodušit nebo vyčíslit.

### 4.1 Zápis a úprava algebraických výrazů

V první řadě je třeba algebraický výraz zapsat tak, aby mu program správně porozuměl. Pomocí Basic Math Assistant můžeme výraz zapsat v příkazovém řádku přímo tak, jak je vysázen v běžném textu. Výhodou je, že nemůže dojít k nedorozumění, v praxi je však zápis tímto způsobem mimořádně pomalý. Kvůli časové úspoře se výrazy zapisují v příkazovém řádku pomocí symbolů vlastních programu *Mathematica* (např. / pro dělení, ^ pro mocninu, Sqrt pro odmocninu, apod.). Pokud máte na počítači nastavenou anglickou klávesnici, je tento zápis velmi rychlý.

Ukažme si oba způsoby zápisu na příkladu vzorce pro výpočet kořene kvadratické rovnice  $x_1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}$ :

$$\text{In[86]:= } \mathbf{x1 = \frac{-b + \sqrt{b^2 - 4ac}}{2a}}$$

$$\text{Out[86]= } \frac{1}{90} (-2 + 2 i \sqrt{89})$$

$$\text{In[87]:= } \mathbf{x1 = (-b + Sqrt[b^2 - 4ac]) / (2a)}$$

$$\text{Out[87]= } \frac{1}{90} (-2 + 2 i \sqrt{89})$$

Při přepisování algebraických výrazů, zejména vyskytují-li se v nich zlomky, je třeba mít na paměti, že počítačový program má násobení a dělení stejnou prioritu a výraz se vyhodnocuje zleva doprava. Výše uvedený vztah  $x_1 = (-b + \sqrt{b^2 - 4ac}) / (2a)$  tedy nemůžeme zapsat ve formě  $x_1 = (-b + \sqrt{b^2 - 4ac}) / 2a$ , protože takto zapsaný algebraický výraz by program pochopil jinak:

$$\text{In[88]:= } \mathbf{x1 = (-b + Sqrt[b^2 - 4ac]) / 2a}$$

$$\text{Out[88]= } \frac{45}{2} (-2 + 2 i \sqrt{89})$$

Jak je vidět z předchozího příkladu, u zlomků musí být v čitateli a jmenovateli výraz v závorkách nejen v případech, že se skládá z několika sčítanců, ale i v případech, kdy obsahuje pouze násobení. Při zapisování výrazů

se proto doporučuje závorky psát velmi pozorně. Ostatně pokud je někde použijete přebytně, na správnost vyhodnocení výrazu to nebude mít vliv.

Základní příkazy na úpravu algebraických výrazů:

<b>Expand</b> [výraz]	upraví výraz do tvaru polynomu
<b>Factor</b> [výraz]	upraví polynom do tvaru součinů a mocnin; součet (nebo rozdíl) zlomků převede na společného jmenovatele
<b>Simplify</b> [výraz]	výraz upraví na co nejjednodušší tvar
<b>Collect</b> [výraz, x]	spojí výrazy se stejnou mocninou x
<b>FactorTerms</b> [výraz, x]	napíše výraz jako součin výrazů, z nichž jeden neobsahuje x
<b>ExpandAll</b> [zlomek]	čitatel i jmenovatel zlomku upraví na polynom
<b>Together</b> [zlomek]	převede všechny zlomky na společný jmenovatel
<b>Cancel</b> [zlomek]	vykrátí zlomek
<b>Apart</b> [zlomek]	upraví zlomek na součet parciálních zlomků

Nyní si uveďme několik příkladů:

In[89]:= **v** = (x + 1) ^ 3 / (x - 1) ^ 2

$$\text{Out[89]} = \frac{(1+x)^3}{(-1+x)^2}$$

In[90]:= **Expand**[v]

$$\text{Out[90]} = \frac{1}{(-1+x)^2} + \frac{3x}{(-1+x)^2} + \frac{3x^2}{(-1+x)^2} + \frac{x^3}{(-1+x)^2}$$

In[91]:= **ExpandAll**[v]

$$\text{Out[91]} = \frac{1}{1-2x+x^2} + \frac{3x}{1-2x+x^2} + \frac{3x^2}{1-2x+x^2} + \frac{x^3}{1-2x+x^2}$$

In[92]:= **Apart**[v]

$$\text{Out[92]} = 5 + \frac{8}{(-1+x)^2} + \frac{12}{-1+x} + x$$

In[93]:= **v1** = 4 x ^ 4 + 36 y ^ 2 + x ^ 3 (12 + 16 y) + x ^ 2 (9 + 48 y + 16 y ^ 2) + x (36 y + 48 y ^ 2)

$$\text{Out[93]} = 4x^4 + 36y^2 + x^3(12 + 16y) + x^2(9 + 48y + 16y^2) + x(36y + 48y^2)$$

In[94]:= **Simplify**[v1]

$$\text{Out[94]} = (3 + 2x)^2 (x + 2y)^2$$

## 4.2 Dosazování proměnných do algebraických výrazů

Při výpočtech s výrazy je časté, že nejdříve „zpracováváme“ vztahy v symbolickém tvaru a teprve následně do výsledného tvaru dosazujeme konkrétní hodnoty proměnných. Vedle konkrétních číselných hodnot můžeme do výrazu dosadit další výrazy. Chceme-li např. ve výrazu dosadit a místo proměnné x, používá se tento příkaz: výraz /. x → a, pokud chceme za hodnoty x1 dosadit a1, za x2 pak a2, atd., použijeme příkaz: výraz /. {x1 →

$a1, x2 \rightarrow a2, \dots$ }. Operátor /. se označuje jako dosazovací operátor, znak  $\rightarrow$  jako tzv. transformační pravidlo. Tento znak se zapisuje na klávesnici jako kombinace znaků  $\rightarrow$ .

Příklad: ve výrazu  $\frac{(1+x)^3}{(-1+x)^2}$  chceme za  $x$  dosadit číslo 3:

```
In[95]:= v =  $\frac{(1+x)^3}{(-1+x)^2}$ 
```

```
Out[95]=  $\frac{(1+x)^3}{(-1+x)^2}$ 
```

```
In[96]:= v /. x -> 3
```

```
Out[96]= 16
```

Budeme-li chtít do stejného výrazu  $\frac{(1+x)^3}{(-1+x)^2}$  za  $x$  dosadit výraz  $\sqrt{y}$ :

```
In[97]:= v /. x -> Sqrt[y]
```

```
Out[97]=  $\frac{(1+\sqrt{y})^3}{(-1+\sqrt{y})^2}$ 
```

Na závěr bychom si měli objasnit rozdíl mezi transformačním pravidlem a přiřazovacím příkazem. Transformační pravidlo  $x \rightarrow 3$  má vliv jen na ten výraz, na který je aplikováno (v našem případě na výraz  $v$ ), nemění ale hodnotu proměnné  $x$ ; přiřazovací příkaz  $x=3$  má obecný charakter a mění trvale hodnotu proměnné  $x$  na 3 (přesněji řečeno: mění hodnotu proměnné  $x$  na 3 dokud ji nezměníme dalším přiřazovacím příkazem, nebo dokud ji nezrušíme příkazem `Clear[x]`). Situaci si přiblížíme na jednoduchém příkladu:

```
In[98]:= v = 3 b
```

```
Out[98]= 6
```

```
In[99]:= v /. b -> 3
```

```
Out[99]= 6
```

```
In[100]:= b
```

```
Out[100]= 2
```

Jak je vidět, dosazení čísla 3 do výrazu  $3b$  umožnilo vyčíslit výraz, ale neodrazilo se ve změně hodnoty proměnné  $b$ . Pokud bychom nyní proměnné  $b$  přiřadili hodnotu 4 (za příkaz jsme dali středník, abychom "ušetřili" řádek s odpovědí):

```
In[101]:= b = 4;
```

pak by už měla proměnná  $b$  trvale hodnotu 4 a případný pokus o to, abychom ji přiřadili ve výrazu  $v$  hodnotu jinou by byl neúspěšný:

```
In[102]:= v /. b -> 3
```

```
Out[102]= 6
```

V praxi se setkáváme i se situacemi, kdy chceme použít stejné transformační pravidlo do většího počtu výrazů. Program *Mathematica* umožňuje označení transformačního pravidla nějakým jménem (s malým počátečním

písmenem), což usnadňuje jeho další používání. Opět si situaci přiblížíme na jednoduchém příkladu. Nejdříve nadefinujeme 2 algebraické výrazy:

$$\text{In[103]:= } \mathbf{v1 = 2 x^3 - 1 x + \frac{1}{2}}$$

$$\text{Out[103]= } \frac{1}{2} - x + 2 x^3$$

$$\text{In[104]:= } \mathbf{v2 = \frac{4 x^3}{5}}$$

$$\text{Out[104]= } \frac{4 x^3}{5}$$

A nyní vytvoříme transformační pravidlo se jménem alois:

$$\text{In[105]:= } \mathbf{alois = x \rightarrow y^5 + 3 y^4 + 2 y^2 + \frac{1}{2} y - 3}$$

$$\text{Out[105]= } x \rightarrow -3 + \frac{y}{2} + 2 y^2 + 3 y^4 + y^5$$

Dosazení výrazu  $-3 + \frac{y}{2} + 2 y^2 + 3 y^4 + y^5$  za x do výrazů v1 a v2 pak bude mít tuto podobu:

$$\text{In[106]:= } \mathbf{v1 /. alois}$$

$$\text{Out[106]= } \frac{7}{2} - \frac{y}{2} - 2 y^2 - 3 y^4 - y^5 + 2 \left( -3 + \frac{y}{2} + 2 y^2 + 3 y^4 + y^5 \right)^3$$

$$\text{In[107]:= } \mathbf{v2 /. alois}$$

$$\text{Out[107]= } \frac{4}{5} \left( -3 + \frac{y}{2} + 2 y^2 + 3 y^4 + y^5 \right)^3$$

## 5 Práce s grafikou v rovině

### 5.1 Obecná pravidla pro práci s grafikou

Práce s grafikou patří mezi velmi silné stránky programu *Mathematica*, systém umožňuje zejména velmi precizní práci s grafy v rovině i v prostoru. Pro geografý je vedle toho užitečná i schopnost programu pracovat s vektorovou grafikou, což umožňuje do programu importovat data v běžných GIS formátech. Program přirozeně není geoinformační, jeho využití je proto spíše v poloze prezentace prostorových geografických dat, než v oblasti jejich analýzy.

Vlastní grafické výstupy programu jsou ukládány v prostorově úsporném vektorovém formátu – tj. jako soubor bodů, linií a ploch a přiřazení, která jim přidělují barvu, tloušťku čar, apod. Body jsou přitom definovány svými souřadnicemi, linie pomocí bodů lomu a plochy pomocí linie, která je ohraničuje. Grafický objekt jako množina dílčích jednoduchých objektů se zobrazuje pomocí funkce `Show`, tu však není třeba použít, pokud je objekt vytvořen pomocí funkcí (např. funkce na zobrazení grafu, 3D grafu, apod.).

Vlastnosti zobrazení grafických objektů (např. tloušťka čar, barva) jsou předdefinovány, je možné je ale měnit dvěma způsoby: lokálně (změní předdefinované parametry pro aktivní dílčí části grafického objektu), nebo globálně (změní předdefinované parametry pro celý objekt). Globální volby vlastností se zadávají ve formě volba  $\rightarrow$  hodnota. Některé grafické volby se provádějí specifikací hodnoty (např. barva), jiné umožňují automatické nastavení optimální hodnoty použitím nastavení `Automatic`.

### 5.2 Základní příkazy

Základní grafické objekty se vytvářejí pomocí těchto příkazů:

<code>Point[{x, y}]</code>	bod se souřadnicemi $[x, y]$
<code>Line[{{x1, y1}, {x2, y2}, ...}]</code>	linie procházející body $[x1, y1], [x2, y2], \dots$
<code>Polygon[{{x1, y1}, {x2, y2}, ...}]</code>	polygon s vrcholy v bodech $[x1, y1], [x2, y2], \dots$
<code>Text[výraz, {x, y}]</code>	text se středem v bodu $[x, y]$

Polygony v podobě základních geometrických útvarů se pak vytvářejí takto:

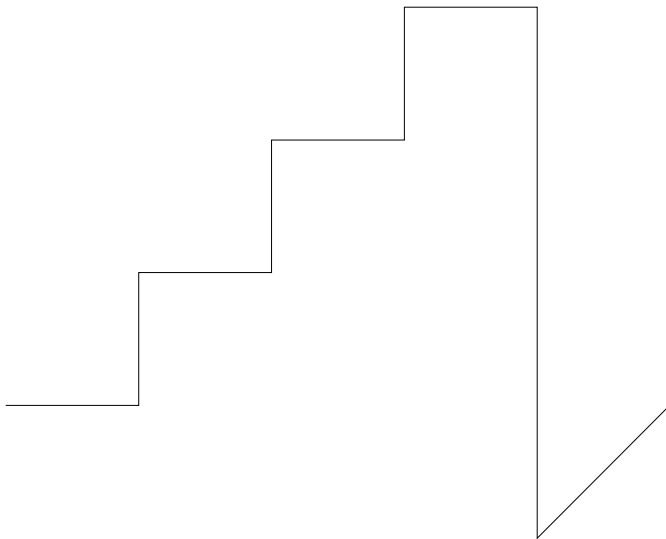
<code>Rectangle[{x1, y1}, {x2, y2}]</code>	obdélník s levým dolním vrcholem $[x1, y1]$ a s pravým horním vrcholem $[x2, y2]$
<code>Circle[{x, y}, r]</code>	kružnice se středem $[x, y]$ a poloměrem $r$
<code>Disk[{x, y}, r]</code>	kruh se středem $[x, y]$ a poloměrem $r$
<code>Circle[{x, y}, {r1, r2}]</code>	elipsa se středem $[x, y]$ a s poloosami $r1$ a $r2$
<code>Circle[{x, x}, r, {uhel1, uhel2}]</code>	oblouk kružnice se středem $[x, y]$ a poloměrem $r$ s počátečním úhlem $uhel1$ a koncovým úhlem $uhel2$ (měřeno v radiánech od kladné osy $x$ proti směru hodinových ručiček)

Jak již bylo uvedeno, grafický objekt se zobrazuje pomocí funkce `Show[Graphic,options]`, skupina objektů pak pomocí `Show[g1,g1,...]`. Postup si ukážeme na jednoduchém příkladě. Nejdříve nadefinujeme 3 různé objekty: lomenou čáru (`cara`), trojúhelník (`trojuhelnik`) a elipsu:

```
In[108]:= cara = Line[{{1, 2}, {2, 2}, {2, 3}, {3, 3},
                    {3, 4}, {4, 4}, {4, 5}, {5, 5}, {5, 1}, {6, 2}, {6, 3}}];
```

```
In[109]:= g1 = Show[Graphics[cara]]
```

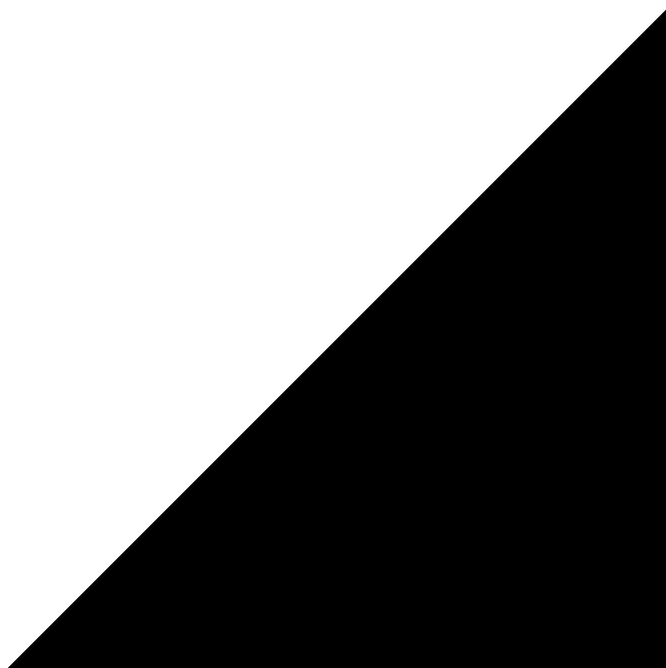
Out[109]=



```
In[110]:= trojuhelnik = Polygon[{{2.5, 2.5}, {5, 5}, {5, 2.5}}];
```

```
In[111]:= g2 = Show[Graphics[trojuhelnik]]
```

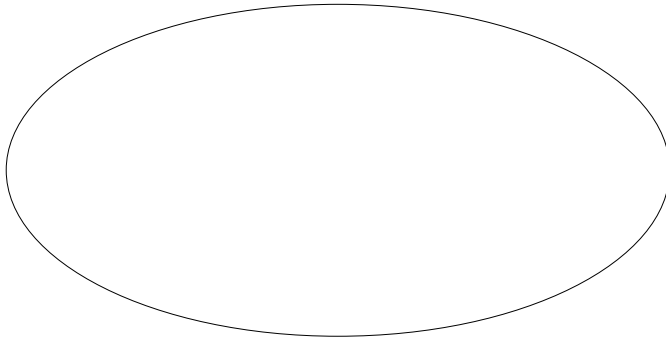
Out[111]=



```
In[112]:= elipsa = Circle[{5, 6}, {2, 1}];
```

In[113]:= **g3 = Show[Graphics[elipsa]]**

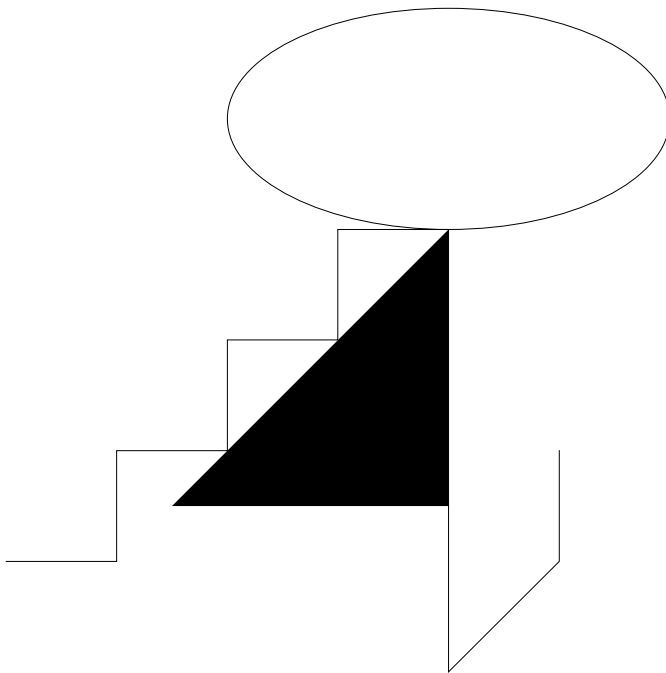
Out[113]=



Pokud budeme chtít všechny 3 objekty zobrazit v jednom obrázku, použijeme příkaz Show[g1,g2,...]:

In[114]:= **Show[g1, g2, g3]**

Out[114]=



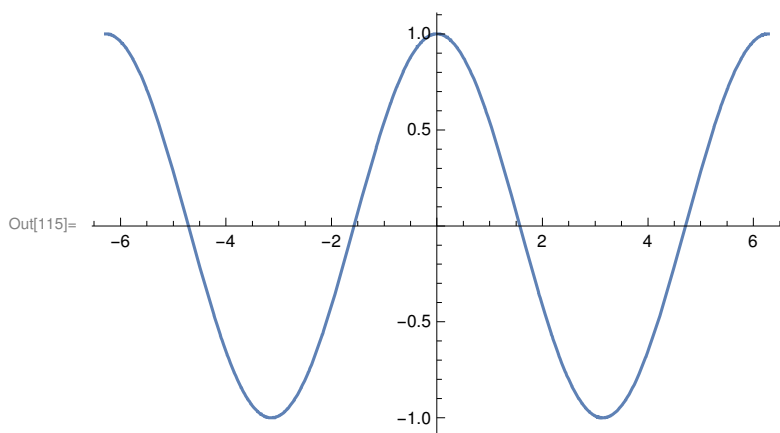
Studenti geografie přirozeně využijí spíše možnosti zobrazovat v programu *Mathematica* grafy.

K jejich zobrazení slouží příkaz Plot, který požadovaný graf okamžitě zobrazí (odpadá tedy mezikrok definování objektu a jeho zobrazení pomocí funkce Show). Základní syntaxe příkazu Plot je:

<b>Plot[f, {x, x<sub>min</sub>, x<sub>max</sub>}</b>	graf funkce f v intervalu < x <sub>min</sub> ; x <sub>max</sub> >
<b>Plot[{f<sub>1</sub>, f<sub>2</sub>, ...}, {x, x<sub>min</sub>, x<sub>max</sub>}</b>	společný graf funkcí f <sub>1</sub> , f <sub>2</sub> ... v intervalu < x <sub>min</sub> ; x <sub>max</sub> >

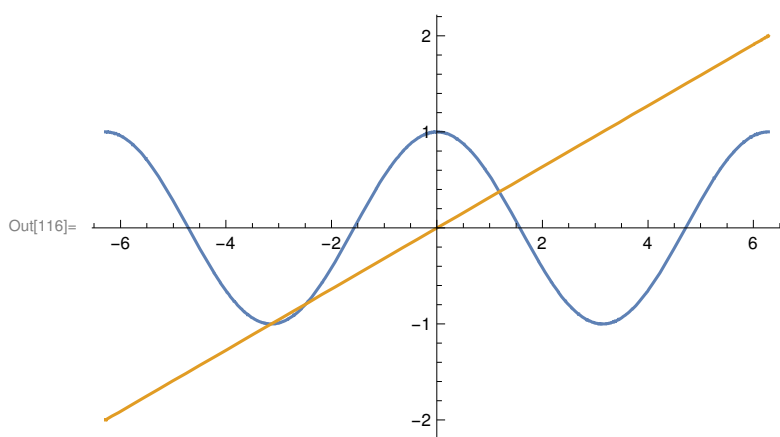
A nyní několik příkladů. Graf funkce cos(x) v intervalu <-2π; 2π>:

In[115]:= `Plot[Cos[x], {x, -2 Pi, 2 Pi}]`



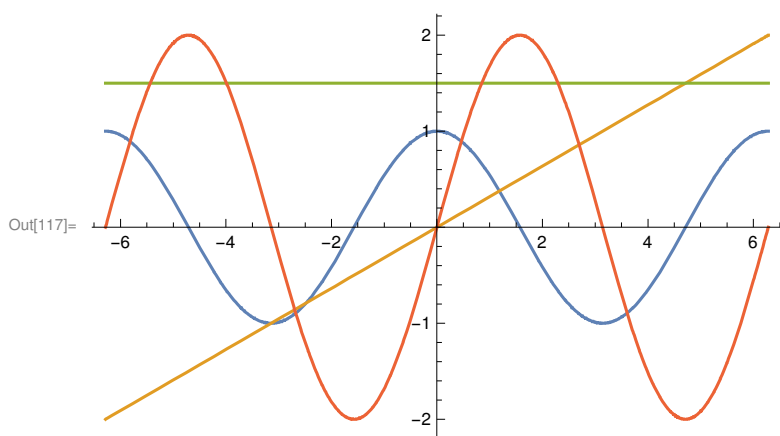
Grafy funkcí  $y = \cos(x)$  a  $y = \frac{x}{\pi}$  v intervalu  $\langle -2\pi; 2\pi \rangle$  v jednom obrázku:

In[116]:= `Plot[{Cos[x], x/Pi}, {x, -2 Pi, 2 Pi}]`



Grafů může být v jednom obrázku podstatně více, doplníme tedy k už zobrazeným ještě  $y = 1.5$  a  $y = 2 \cos(x - \frac{\pi}{2})$ :

In[117]:= `Plot[{Cos[x], x/Pi, 1.5, 2 Cos[x - Pi/2]}, {x, -2 Pi, 2 Pi}]`



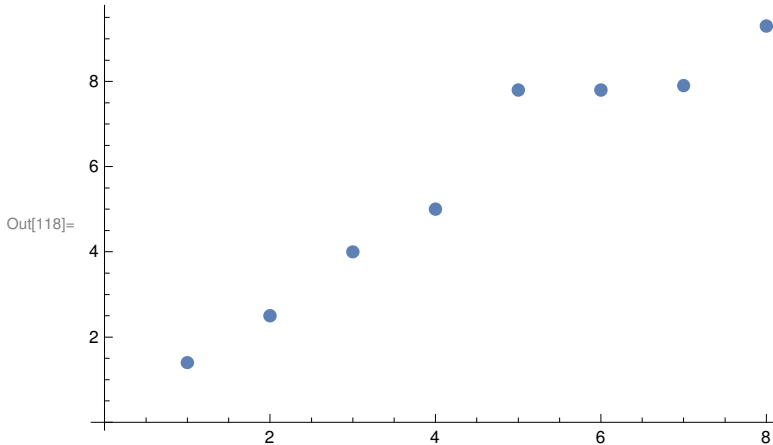
Ke zobrazení bodového grafu slouží příkaz `ListPlot`, který požadovaný graf okamžitě zobrazí (odpadá tedy opět funkce `Show`). Základní syntaxe příkazu `ListPlot` je:



<code>ListPlot[{y<sub>1</sub>, y<sub>2</sub>, ...}]</code>	zobrazí body [1, y <sub>1</sub> ], [2, y <sub>2</sub> ], ...
<code>ListPlot[{{x<sub>1</sub>, y<sub>1</sub>}, {x<sub>2</sub>, y<sub>2</sub>}, ...]</code>	zobrazí body [x <sub>1</sub> , y <sub>1</sub> ], [x <sub>2</sub> , y <sub>2</sub> ], ...

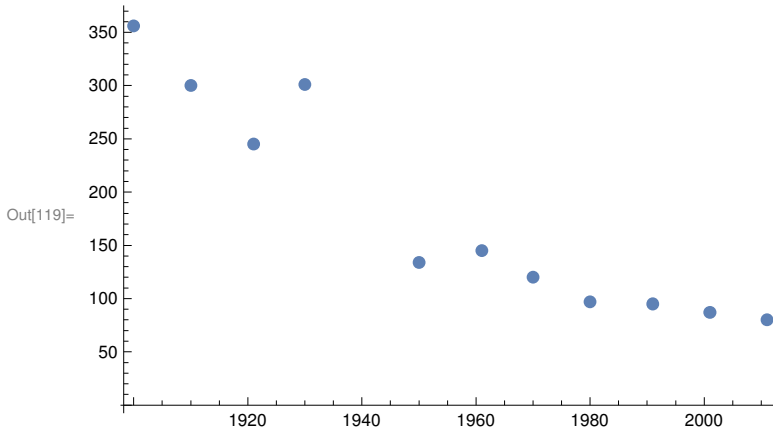
První typ se hodí pro veličiny, které mají na x-ové souřadnici pravidelné rozestupy (např. volební výsledky strany ČSSD v obcích seřazené podle velikosti). Příklad:

```
In[118]:= ListPlot[{1.4, 2.5, 4, 5, 7.8, 7.8, 7.9, 9.3}]
```



Druhý typ je vhodný pro případy, kdy jsou data na x-ové souřadnici vzdálena nerovnoměrně - např. údaje o počtu obyvatel za sčítání, která nejsou v pravidelných intervalech:

```
In[119]:= ListPlot[{{1900, 356}, {1910, 300}, {1921, 245}, {1930, 301}, {1950, 134},
  {1961, 145}, {1970, 120}, {1980, 97}, {1991, 95}, {2001, 87}, {2011, 80}}]
```



### 5.3 Změny stylů kreslení

Pro změnu stylů komunita uživatelů zpravidla používá textových příkazů, pro začátečníky ale může být poměrně vítanou pomocí i nabídka v Suggestions Bar (řádek se objeví pod grafem po jeho vykreslení), který umožňuje základní úpravy bez znalosti přesné syntaxe příkazů. Pro specifikaci barvy se nejčastěji používají příkazy:

<b>GrayLevel</b> [i]	odstín šedé mezi černou (hodnota 0) a bílou (hodnota 1)
<b>RGBColor</b> [r, g, b]	barva definována barevným modelem RGB (r – červená, g – zelená a b – modrá), mohutnost složek se udává od 0 do 1
<b>CMYKColor</b> [c, m, y, k]	barva definována barevným modelem CMYK (c – azurová, m – purpurová, y – žlutá, k – černá), mohutnost složek se udává od 0 do 1
<b>Hue</b> [h]	Odstín (0 – černá, 1 – bílá)
<b>Hue</b> [h, s, b]	Odstín (h), sytost (s) a jas (b)

Velikost bodů se mění pomocí příkazů:

<b>PointSize</b> [r]	body se vykreslí jako kruhy s poloměrem r (relativní veličina od 0 do 1, vzhledem k velikosti zobrazovaného prostoru)
<b>AbsolutePointSize</b> [d]	body se vykreslí jako kruhy s průměrem d (absolutní veličina bez ohledu na velikost zobrazovaného prostoru)

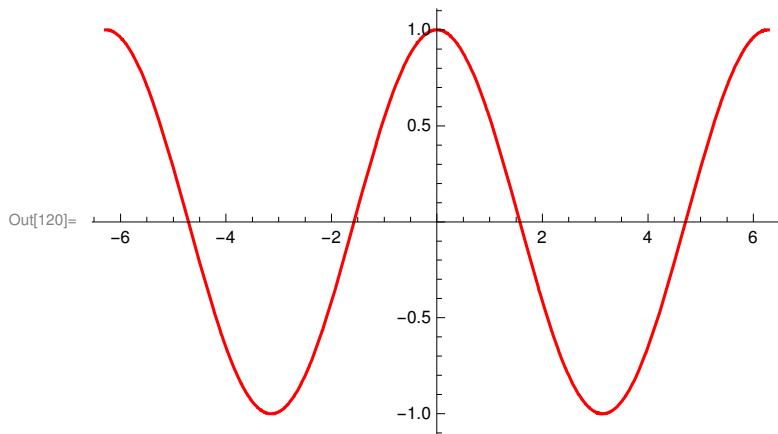
Tvar a tloušťka čar:

<b>Thickness</b> [r]	čárka s tloušťkou r (relativní veličina od 0 do 1, vzhledem k velikosti zobrazovaného prostoru)
<b>AbsoluteThickness</b> [d]	čárka s tloušťkou d (absolutní veličina)
<b>Dashing</b> [{r <sub>1</sub> , r <sub>2</sub> , ...}]	čárkovaná / čerchovaná čára (r <sub>1</sub> , r <sub>2</sub> udávají relativní délky čar a mezer)

Samotné změny nebo úpravy stylu se provádějí pomocí příkazu `PlotStyle` → {seznam příkazů}, pokud jde o jeden graf, případně `PlotStyle` → {{seznam příkazů pro 1. graf},{seznam příkazů pro 2. graf},{seznam příkazů pro 3. graf},...}, pokud se jedná o více grafů v jednom grafickém objektu. Barva pozadí se specifikuje příkazem `Background` → barva, barva čar pak příkazem `DefaultColor` → barva. Barvu pak označíme některým ze způsobů uvedených v přehledu příkazů pro specifikaci barvy. Tyto příkazy je vhodné umisťovat za údaje o funkci a intervalu do hranaté závorky příkazu `Plot`, tj. `Plot[f, {x, xmin, xmax}, PlotStyle → {seznam příkazů}]`

Způsob provádění úprav stylu grafů asi nejlépe přiblíží několik příkladů. Například pokud chceme, aby byla barva grafu červená:

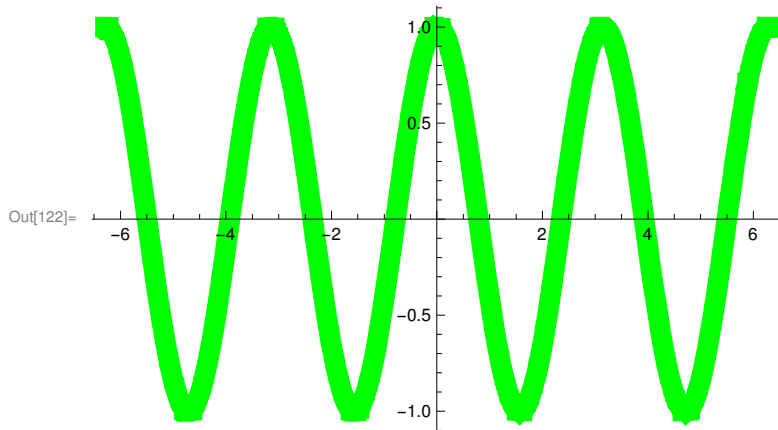
```
In[120]:= Plot[Cos[x], {x, -2 Pi, 2 Pi}, PlotStyle → RGBColor[1, 0, 0]]
```



In[121]=

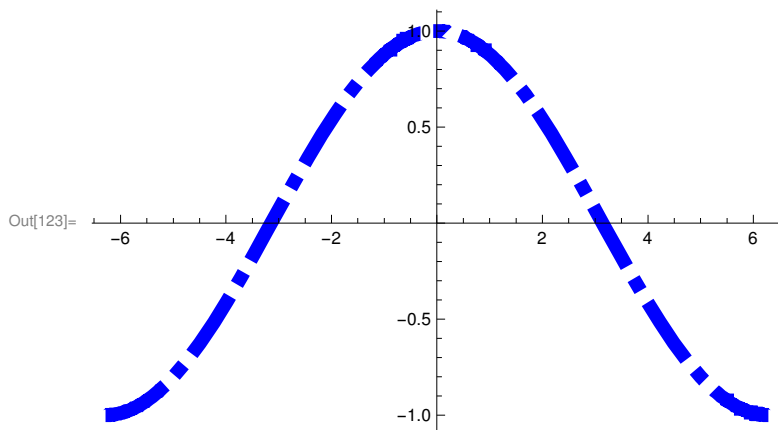
... nebo pokud má být graf vykreslen silnější zelenou linií:

In[122]= `Plot[Cos[2 x], {x, -2 Pi, 2 Pi}, PlotStyle -> {RGBColor[0, 1, 0], Thickness[0.03]}]`



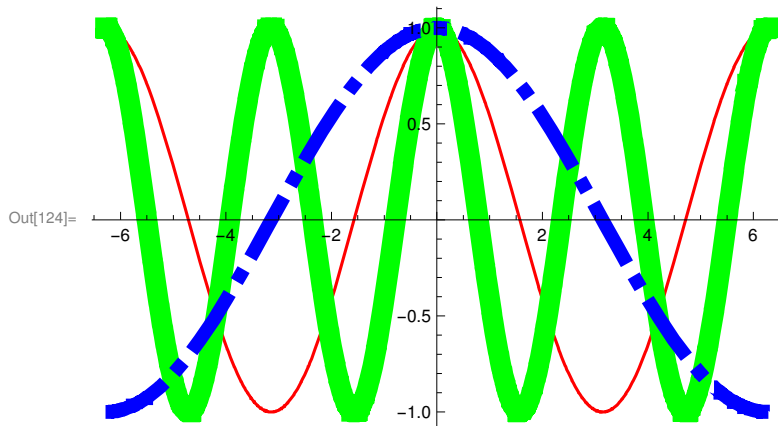
... případně čerchovanou jen mírně silnější modrou linií:

In[123]= `Plot[Cos[x / 2], {x, -2 Pi, 2 Pi}, PlotStyle -> {RGBColor[0, 0, 1], Thickness[0.02], Dashing[{0.1, 0.02, 0.02, 0.02]}]]`



Pokud bychom chtěli všechny 3 předchozí grafy vykreslit v jednom obrázku, byl by už příkaz s instrukcemi programu *Mathematica* poměrně dlouhý:

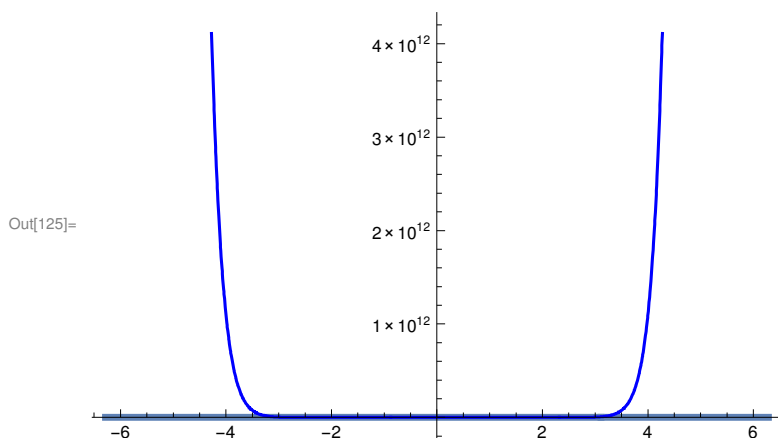
```
In[124]:= Plot[{Cos[x], Cos[2 x], Cos[x / 2]}, {x, -2 Pi, 2 Pi},
  PlotStyle -> {{RGBColor[1, 0, 0]}, {RGBColor[0, 1, 0], Thickness[0.03]},
  {RGBColor[0, 0, 1], Thickness[0.02], Dashing[{0.1, 0.02, 0.02, 0.02]}}}]
```



## 5.4 Úprava zobrazované plochy grafu

Při zobrazování grafu vybírá program *Mathematica* zobrazovanou oblast tak, aby byl graf “ještě” dobře čitelný. V praxi, zejména v případech, kdy byl nevhodně zvolen interval zobrazovaných funkcí, to znamená, že některé příliš vzdálené body mohou ležet mimo zobrazovanou plochu grafu, nebo že i přes snahu programu je graf nepřehledný. Např. pokud budeme chtít zobrazit funkce  $\cos(x)$  a  $x^{20}-10$  v jenom grafu, bude výsledek opravdu podivný:

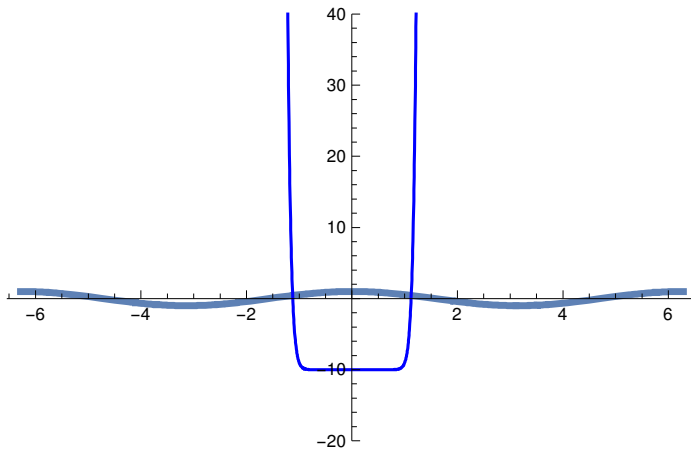
```
In[125]:= Plot[{Cos[x], x^20 - 10}, {x, -2 Pi, 2 Pi},
  PlotStyle -> {Thickness[0.01], RGBColor[0, 0, 1]}}
```



V takovýchto případech je užitečné použití příkazu `PlotRange`, který může omezit rozsah zobrazované oblasti - buď jen určitým rozsahem hodnot na ose y (`PlotRange -> {y_min, y_max}`), nebo jak na ose y, tak na ose x (`PlotRange -> {{x_min, x_max}, {y_min, y_max}}`). Například v předchozím grafu omezíme zobrazované hodnoty na ose y na interval od -2 do +40, což by mělo umožnit lepší srovnání obou funkcí:

```
In[126]:= Plot[{Cos[x], x^20 - 10}, {x, -2 Pi, 2 Pi},
  PlotStyle -> { Thickness[0.01], RGBColor[0, 0, 1]}, PlotRange -> {-20, 40}]
```

Out[126]=



## 5.5 Úprava os grafu a oblasti grafu

Příkazy pro zobrazování a popis os grafu:

<b>Axes → True</b>	osy se zobrazí (tato hodnota je předdefinována)
<b>Axes → False</b>	osy se nezobrazí
<b>Axes → {True, False}</b>	x – ová osa se zobrazí, y – ová osa ne
<b>AxesOrigin → {x, y}</b>	osy se protnou v bodě [x, y] (předdefinována je hodnota Automatic, tj. průsečík je nastaven tak, aby se zobrazila co největší část grafu)
<b>AxesStyle → {příkaz1, příkaz2, ...}</b>	nastavení stylu kreslení os
<b>AxesStyle → {{p1, p2, ...}, {pA, pB, ...}}</b>	nastavení různého stylu pro osu x a y
<b>AxesLabel → None</b>	osy bez popisu (předdefinovaná hodnota)
<b>AxesLabel → " text "</b>	popis osy y
<b>AxesLabel → {" text1 ", " text2 "}</b>	popis os x a y
<b>Ticks → None</b>	značky na osách se nezobrazí (předdefinovaná hodnota je Automatic – tj. polohu značek vypočítá Mathematica)
<b>Ticks → {{pravidlo}, {pravidlo}} ... {x1, x2, ...}</b>	specifické umístění značek na ose x a y ... umístění značek na specifické pozice

<code>... {{x1, "text1"}, {x2, "text2"}, ...}</code>	... umístění značek na specifické pozice a jejich popis příslušným textem
<code>... {{x1, "text1", délka1}, {x2, "text2", délka2}, ...}</code>	... umístění značek specifikované délky na specifické pozice a jejich popis příslušným textem
<code>... {{x1, text1, {délkakladná, délkazáporná}}, ...}</code>	... umístění značek specifikované délky v kladném a jiné délky v záporném směru na specifické pozice a jejich popis příslušným textem
<code>... {{x1, "text1", délka1, styl1}, ...}</code>	... umístění značek specifikované délky a stylu na specifické pozice a jejich popis příslušným textem

Příkazy pro zobrazování a úpravy rámečku:

<b>Frame</b> → <b>True</b>	okolo obrázku se vykreslí rámeček (hodnoty x a y jsou pak uvedeny na rámečku, nikoliv na osách)
<b>Frame</b> → <b>False</b>	rámeček se nezobrazí (předdefinovaná hodnota)
<b>FrameStyle</b> → {příkaz1, příkaz2, ...}	specifikace stylu rámečku
<b>FrameStyle</b> → {{příkazy1}, {příkazy2}}	specifikace stylu pro každou stranu rámečku samostatně (v pořadí dolní, levá)
<b>FrameLabel</b> → <b>None</b>	rámeček není označen (předdefinovaná hodnota)
<b>FrameLabel</b> → {"text1", "text2", ...}	označení každé strany rámečku
<b>RotateLabel</b> → <b>True</b>	svislé osy / hrany jsou označeny zezdola nahoru (předdefinovaná hodnota)
<b>RotateLabel</b> → <b>False</b>	všechny osy / hrany jsou označeny vodorovně
<b>FrameTicks</b> → <b>Automatic</b>	na hranách rámečku jsou značky, jejich poloha se vypočítává automaticky (předdefinovaná hodnota)
<b>FrameTicks</b> → <b>None</b>	na hranách rámečku nejsou značky

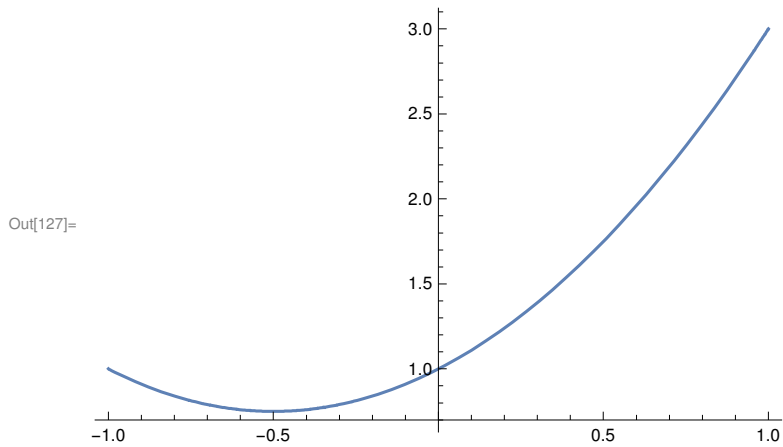
Příkazy pro zobrazování a úpravy mřížky:

<b>GridLines</b> → <b>None</b>	mřížka se nezobrazí (předdefinovaná možnost)
<b>GridLines</b> → <b>Automatic</b>	zobrazí se mřížka podle vnitřního algoritmu programu
<b>GridLines</b> → {{hodnoty x}, {hodnoty y}}	zobrazí se mřížka v určených hodnotách

Opět si ukážeme několik příkladů. Jak již bylo uvedeno v tabulce příkazů, automaticky se nastavuje průsečík

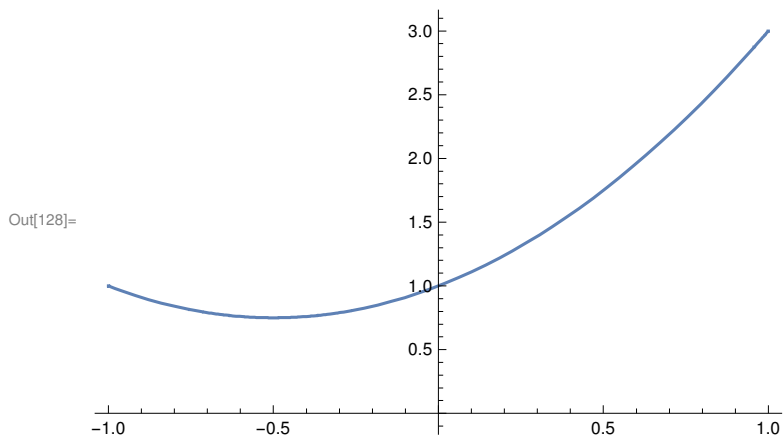
os tak, aby byla zobrazena co největší část grafu. To ale může někdy vést k matoucím závěrům např. o kořenech rovnice. Jako příklad si uveďme automatickou volbu grafu funkce  $y=x^2+x+1$ :

In[127]:= `Plot[x^2 + x + 1, {x, -1, 1}]`



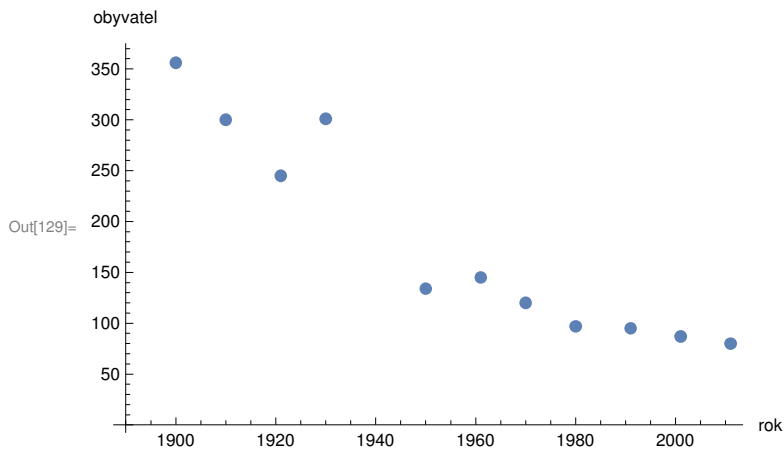
Pokud program *Mathematica* “přinutíme” umístit průsečík os do bodu  $[0;0]$ , je situace mnohem jasnější:

In[128]:= `Plot[x^2 + x + 1, {x, -1, 1}, AxesOrigin -> {0, 0}]`



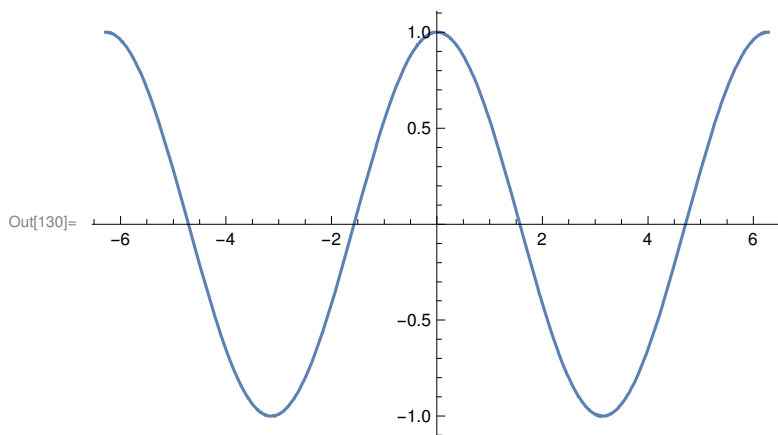
Popis os grafu si ukážeme na jednom z předchozích příkladů:

```
In[129]:= ListPlot[{{1900, 356}, {1910, 300}, {1921, 245}, {1930, 301}, {1950, 134},
  {1961, 145}, {1970, 120}, {1980, 97}, {1991, 95}, {2001, 87}, {2011, 80}},
  PlotStyle -> PointSize[0.02], AxesLabel -> {rok, obyvatel}, AxesOrigin -> {1890, 0}]
```



Na ukázkou manipulace s polohou značek na ose si zopakujeme automatické vykreslení grafu funkce  $y=\cos(x)$ :

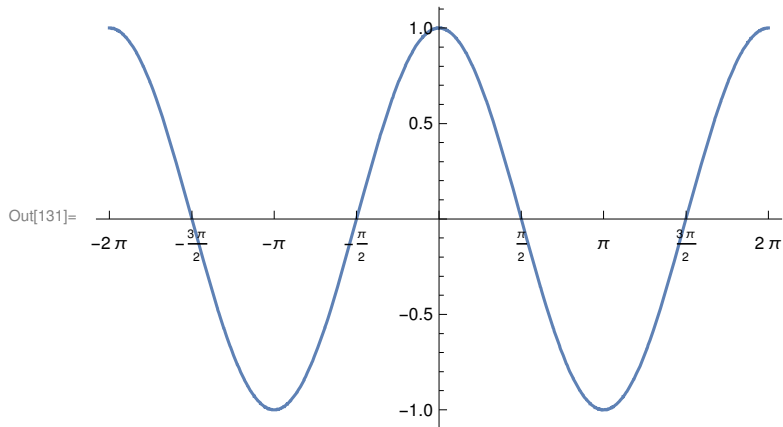
```
In[130]:= Plot[Cos [x], {x, -2 Pi, 2 Pi}]
```



Popis osy x je velmi nepřehledný a také “nenázorný”, přitom lze mnohem lepšího efektu dosáhnout poměrně jednoduchou úpravou:



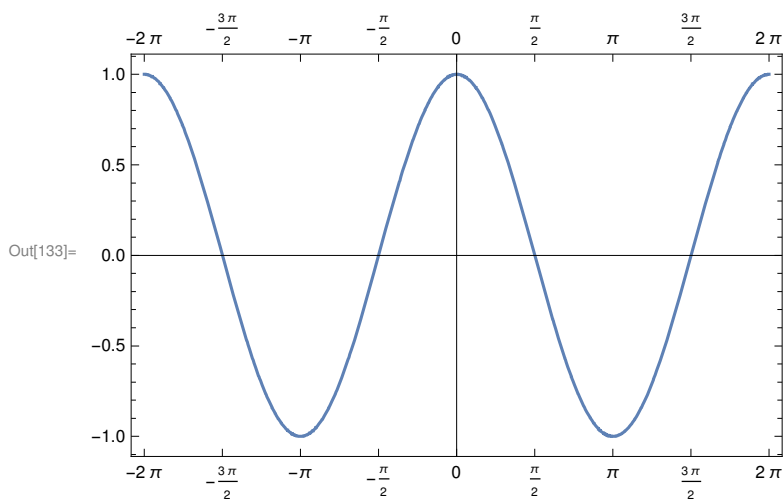
```
In[131]:= Plot[Cos[x], {x, -2 Pi, 2 Pi},
  Ticks → {{-2 Pi, -3/2 Pi, -Pi, -1/2 Pi, 0, Pi/2, Pi, 3/2 Pi, 2 Pi}, Automatic}]
```



```
In[132]:=
```

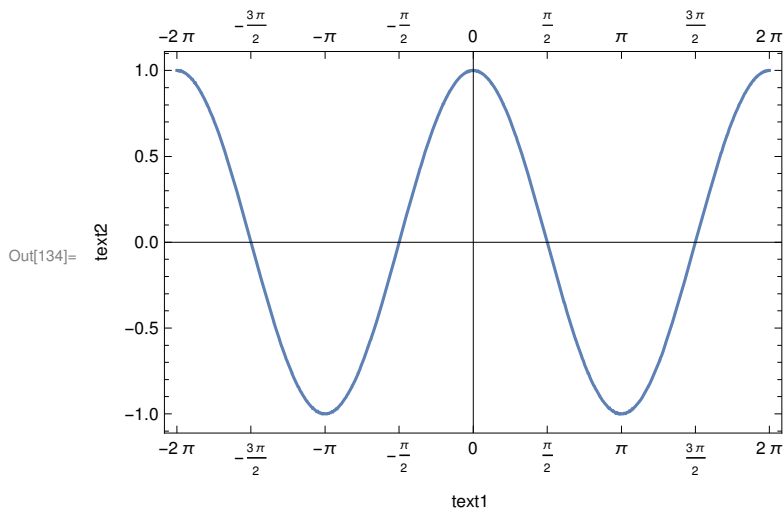
Stejný graf v “rámečkové úpravě” pak vypadá takto:

```
In[133]:= Plot[Cos[x], {x, -2 Pi, 2 Pi}, Frame → True, FrameTicks →
  {{-2 Pi, -3/2 Pi, -Pi, -1/2 Pi, 0, Pi/2, Pi, 3/2 Pi, 2 Pi}, Automatic}]
```



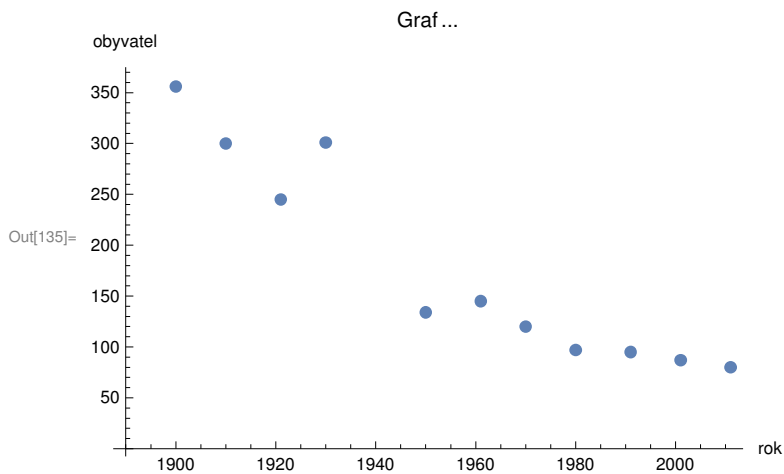
Popis hran rámečku můžeme provést pomocí `FrameLabel` → {“text1 “, “text2”, ...}:

```
In[134]:= Plot[Cos [x], {x, -2 Pi, 2 Pi}, Frame → True, FrameTicks →
  {{-2 Pi, -3 / 2 Pi, -Pi, -1 / 2 Pi, 0, Pi / 2, Pi, 3 / 2 Pi, 2 Pi }, Automatic},
  FrameLabel → {"text1 ", "text2"}]
```



Název grafu může umístit do jeho obrázku volbou PlotLabel → “popis“:

```
In[135]:= ListPlot[{{1900, 356}, {1910, 300}, {1921, 245}, {1930, 301}, {1950, 134},
  {1961, 145}, {1970, 120}, {1980, 97}, {1991, 95}, {2001, 87}, {2011, 80}},
  PlotStyle → PointSize[0.02], AxesLabel → {rok, obyvatel},
  AxesOrigin → {1890, 0}, PlotLabel → Graf ...]
```



## 6 Práce s 3D grafikou

### 6.1 Obecná pravidla pro práci s 3D grafikou

Podobně jako v případě 2D grafiky jsou vlastní grafické výstupy programu ukládány v prostorově úsporném vektorovém formátu – tj. jako soubor bodů, linií, ploch a v tomto případě i prostorových objektů a přiřazení, která jim přidělují barvu, tloušťku čar, apod. Body jsou přitom definovány svými souřadnicemi, linie pomocí bodů lomu, plochy pomocí linií, která je ohraničuje a prostorové objekty pomocí ploch, které je ohraničují. Grafický objekt jako množina dílčích jednoduchých objektů se zobrazuje i v tomto případě pomocí funkce `Show`, v tomto případě v podobě `Show[Graphics3D[objekt]]`.

### 6.2 Základní příkazy a souřadnicový systém pro 3D grafiku

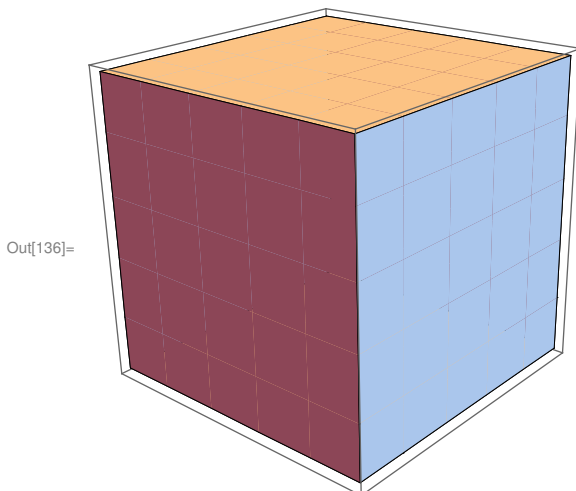
Základní grafické objekty se vytvářejí pomocí těchto příkazů:

<code>Point[{x, y, z}]</code>	bod se souřadnicemi $[x, y, z]$
<code>Line[{{x1, y1, z1}, {x2, y2, z2}, ...}]</code>	linie procházející body $[x1, y1, z1]$ , $[x2, y2, z2], \dots$
<code>Polygon[{{x1, y1, z1}, {x2, y2, z2}, ...}]</code>	polygon s vrcholy v bodech $[x1, y1, z1]$ , $[x2, y2, z2], \dots$
<code>Text[výraz, {x, y, z}]</code>	text od bodu se souřadnicemi $[x, y, z]$

Základní prostorový útvar, tzv. “jednotková” krychle, se vytváří příkazem `Cuboid[{x,y,z}]`. Jde o krychli s dolním levým vrcholem  $[x, y, z]$  a horním pravým vrcholem  $[x+1, y+1, z+1]$ .

Příklad:

```
In[136]:= Show[Graphics3D[Cuboid[{0, 0, 0}]]]
```



Při práci s 3D grafikou se používají shodné příkazy jako při práci s 2D grafikou (např. `PointSize`, `Thickness`,

Dashing - pokud jde o relativní jednotky, resp. AbsolutePointSize, AbsoluteThickness, AbsoluteDashing, pokud vyjadřujeme údaje v jednotkách absolutních), přirozeně je ale nutné v syntaxi počítat s třetí osou z. Shodně s 2D grafikou se také vyjadřuje v 3D grafice barva.

Protože se při zobrazování 3D objektů snaží *Mathematica* navodit co nejpřesvědčivější prostorový efekt, simuluje program účinky osvětlení objektu. Tuto funkci je možné vypnout nebo opět zapnout pomocí volby `Lighting→False` a `Lighting→True`. Speciálně pro prostorovou grafiku jsou určeny také příkazy `EdgeForm []` (zruší vykreslování čar na hranách polygonu), `EdgeForm [grafické pravidlo]` (vykreslí čáry požadovaným způsobem), `FaceForm [grafické pravidlo 1, grafické pravidlo 2]` (přední stěny polygonu vybarví podle pravidla 1, zadní podle pravidla 2).

Při automatickém nastavení se prostorové objekty vykreslují do nitra pomyslného hranolu (box). Všechny části objektu, které leží mimo tento hranol pak nejsou zobrazeny. Velikost hranolu určí program tak, aby podstatná část objektu byla zobrazena, nicméně podobně jako ve 2D grafech mohou chybět v zobrazení extrémní nebo podle programu “nezajímavé” hodnoty. Box se v grafech zobrazuje, je ale možné ho potlačit příkazem `Boxed→False` (či obnovit příkazem `Boxed→True`). Ve většině případů potlačení hran vede k částečné ztrátě prostorového dojmu.

Rozsah zobrazované oblasti lze upravit podobně jako ve 2D grafech příkazem `PlotRange→{zmin, zmax}`; po jeho provedení se zobrazí jen ty části grafu, které jsou v zadaném rozmezí souřadnic z. Obnova automatického výpočtu zobrazované oblasti se provede příkazem `PlotRange→Automatic`, všechny části objektu se zobrazí na základě žádosti ve tvaru `PlotRange→All`. U hranolu, ve kterém je průběh 3D funkcí zobrazován, je také možno změnit programem vypočtený poměr délky stran. Používá se k tomu příkaz `BoxRatios→{a, b, c}`, x-ová, y-ová a z-ová hrana jsou pak nastaveny v poměru a : b : c.

Další hodnota, kterou lze nastavit, je bod, ze kterého objekt pozorujeme. To lze volbou `ViewPoint→{p,q,r}`, která zajistí, že objekt je zobrazován z pohledu bodu o souřadnicích [p,q,r]. Nejde přitom o souřadnice vztahované k osám x, y a z, ale o hodnoty souřadnic ve zvláštním systému souřadnic se středem uprostřed hranolu. Změna místa, ze kterého je objekt pozorován, je v praxi výrazně zjednodušena použitím interaktivního menu programu. V programu lze nastavit i změnu středu zobrazení (`ViewCenter`) a orientace boxu (`ViewVertical`).

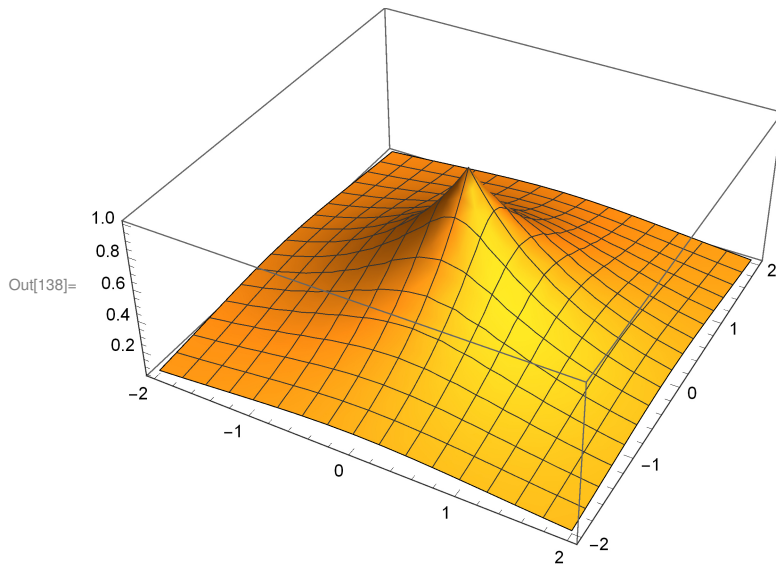
Ukázka:

Zobrazení grafu funkce  $z=e^{-\sqrt{x^2+y^2}}$  :

```
In[137]:= z = Exp[-Sqrt[x^2 + y^2]]
```

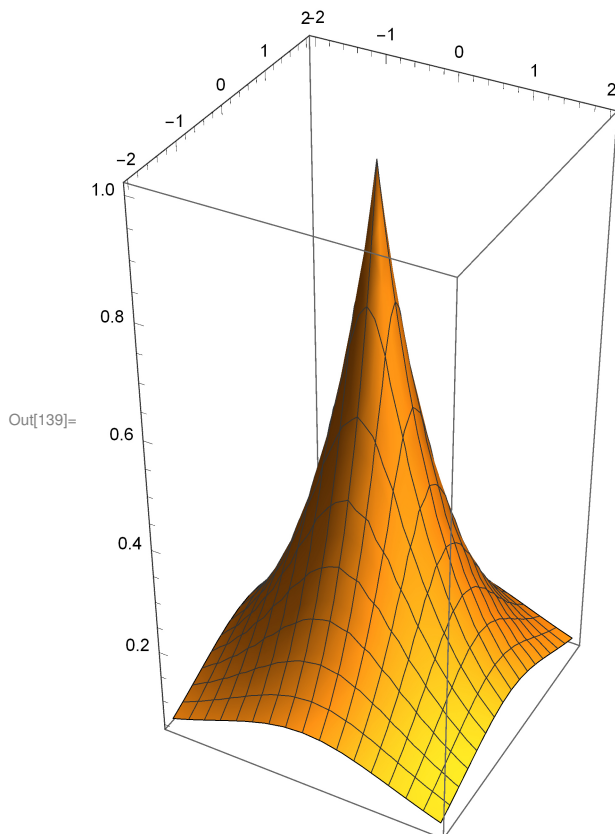
```
Out[137]:= e-√x2+y2
```

```
In[138]:= graf1 = Plot3D[Exp[-Sqrt[x^2 + y^2]], {x, -2, 2}, {y, -2, 2}]
```



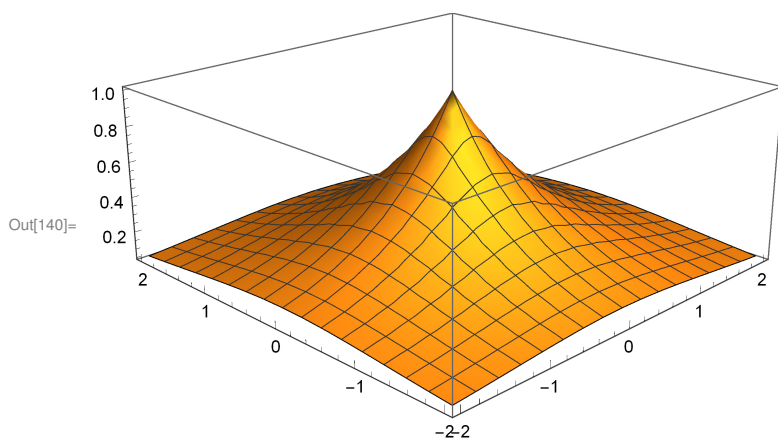
změna poměru stran boxu na 1 : 1 : 2:

```
In[139]:= Show[graf1, BoxRatios -> {1, 1, 2}]
```



Pohled z jiného úhlu:

```
In[140]:= Show[graf1, ViewPoint → {-2, -2, 1}]
```



### 6.3 Úpravy 3D grafů

Možnosti úpravy 3D grafů jsou obdobné, jako v případě 2D grafů. Rovněž v tomto případě lze měnit styl a popisky os, podobu zobrazované oblasti, barvy a další parametry. Příkazy jsou obdobné jako u 2D grafů, úplný seznam lze nalézt v nápovědě programu.

## 7 Specifické typy grafů využívané v geografii

*Mathematica* umožňuje i tvorbu grafů, které jsou ve větší míře využívány pouze v geografii. Některé z nich si ukážeme v této kapitole.

### 7.1 Trojúhelníkový graf

Trojúhelníkový graf slouží k prezentaci jednotek, u kterých sledujeme jev mající tři souřadnice, jejichž součtem dostáváme 1 (nebo 100 %). Například sledujeme v územních jednotkách zaměstnanost v sektorech hospodářství, zmíněné tři souřadnice představují zaměstnanost v primárním sektoru, sekundárním a terciárním (v součtu 100 % zaměstnaných). Takováto data lze jednoduše graficky prezentovat, konkrétně prostřednictvím tzv. trojúhelníkového grafu. Každá jednotka je zobrazena prostřednictvím jednoho bodu v rovnostranném trojúhelníku, jehož strany jsou nositelkami stupnic. Zobrazený bod A [20; 55; 25] tak představuje územní jednotku se zaměstnaností I. – 20 %; II. – 55 %; III. – 25 %.

## Vytvoření trojúhelníkového grafu:

Trojúhelníkový graf není v programu *Mathematica* předdefinován. Jeho vytvoření je už poněkud složitější záležitostí, jak je ostatně vidět z definice funkce, která ho dokáže vytvořit:

```
In[141]:= TrianglePlot[data_List, {min_, max_},
  {navezA_String, navezB_String, navezC_String}, legend_: False] :=
Module[{body, skala},
  skala[s_] := promennaA s + promennaB /. Solve[ {.025 == promennaA min + promennaB,
    .1 == promennaA max + promennaB}, {promennaA, promennaB}];
  If[MemberQ[data, _RGBColor, 2],
    (*multiColor*)
    body = Map[ {Last[#], Drop[#, -1] /. {a_Real, b_Real, c_Real, size_?NumberQ} =>
      {PointSize[First@skala[size]],
        Point[ { { 1/2 (2 b + c) / (a + b + c), sqrt(3) c / (2 (a + b + c)) } } ] &, data},
      (*defaultColor*)body = data /. {a_Real, b_Real, c_Real, size_?NumberQ} =>
      {PointSize[First@skala[size]], Point[ { { 1/2 (2 b + c) / (a + b + c), sqrt(3) c / (2 (a + b + c)) } } ]
    ];

  Graphics[ {
    (*triangle*)Line[ { {0, 0}, {1, 0}, {1/2, sqrt(3)/2}, {0, 0} } ], Gray,
    Line[Table[ { {x, 0},
      {1 - Cos[60 Degree] * (1 - x), Sin[60 Degree] * (1 - x)} }, {x, .1, .9, .1} ]],
    Line[Table[ { {x, 0}, {Cos[60 Degree] * x, Sin[60 Degree] * x} },
      {x, .1, .9, .1} ]],
    Line[Table[ { {Cos[60 Degree] * x, Sin[60 Degree] * x},
      {1 - Cos[60 Degree] * (x), Sin[60 Degree] * (x)} }, {x, .1, .9, .1} ]],
```

```

(*axesNumber*)Black, Table[
  Text[Style[Round[x * 100], FontFamily → "Arial"], {x, -.04}], {x, 0, 1, .2}],
Table[Rotate[Text[Style[Round[x * 100], FontFamily → "Arial"],
  {1 - Cos[60 Degree] * x + Cos[30 Degree] * .04,
  Sin[60 Degree] * x + Sin[30 Degree] * .04}], -π / 3], {x, 0, 1, .2}],
Table[Rotate[Text[Style[Round[100 - x * 100], FontFamily → "Arial"],
  {Cos[60 Degree] * x - Cos[30 Degree] * .04,
  Sin[60 Degree] * x + Sin[30 Degree] * .04}], π / 3], {x, 0, 1, .2}],

(*data*)body,

(*axesNames*)
Rotate[Text[Style[nazevA, 14], {Cos[60 Degree] * .5 - Cos[30 Degree] * .1,
  Sin[60 Degree] * .5 + Sin[30 Degree] * .1}], π / 3],
Text[Style[nazevB, 14], {.5, -.1}],
Rotate[Text[Style[nazevC, 14], {1 - Cos[60 Degree] * .5 + Cos[30 Degree] * .1,
  Sin[60 Degree] * .5 + Sin[30 Degree] * .1}], -π / 3],

If[legend, {LightGray, EdgeForm[Black], Disk[ {.1, -.2 + .0875 / 2}, .0875 / 2],
  Disk[ {.22, -.2 + .0625 / 2}, .0625 / 2], Disk[ {.32, -.2 + .0375 / 2}, .0375 / 2],
  Black, Text[Style[min +  $\frac{5(\max - \min)}{6}$ , FontFamily → "Arial"], {.1, -.23}],
  Text[Style[min +  $\frac{\max - \min}{2}$ , FontFamily → "Arial"], {.22, -.23}],
  Text[Style[min +  $\frac{(\max - \min)}{6}$ , FontFamily → "Arial"], {.32, -.23}]}]]
]]
]

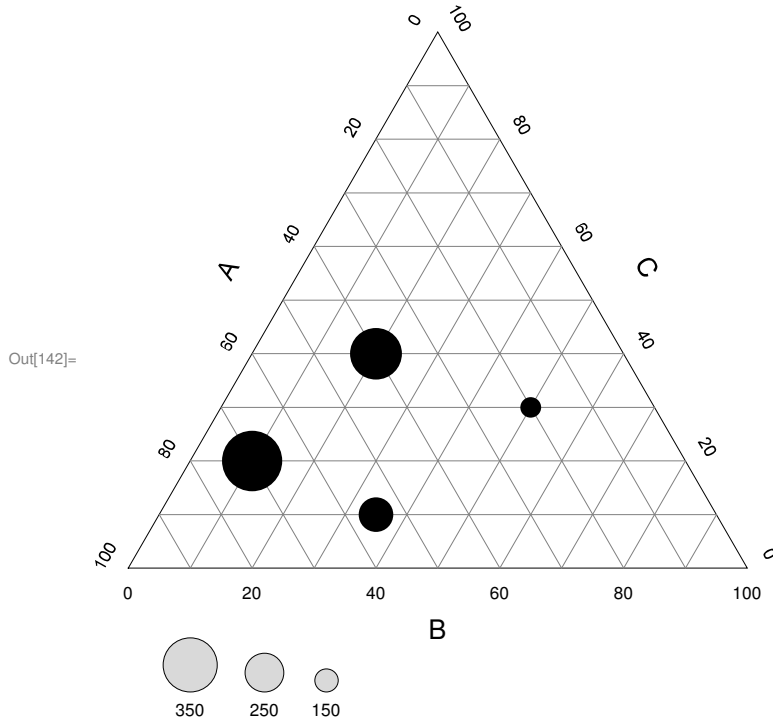
```

V takto nadefinovaném grafu

(TrianglePlot[data\_List, {min\_, max\_}, {nazevA\_String, nazevB\_String, nazevC\_String}, legend\_:False]) znamená jednotlivé údaje: data\_list je soubor dat ve formátu {souřadnice a1, souřadnice b1, souřadnice c1, hodnota}, {souřadnice a2, souřadnice b2, souřadnice c2, hodnota2}, ... , {min\_, max\_} jsou meze hodnot pro tvorbu legendy, {nazevA\_String, nazevB\_String, nazevC\_String} je soubor popisků os a, b a c a konečně na posledním místě je informace, jestli má (True) nebo nemá (False) být uvedena legenda grafu. Ukážeme si použití funkce na jednoduchém souboru dat:

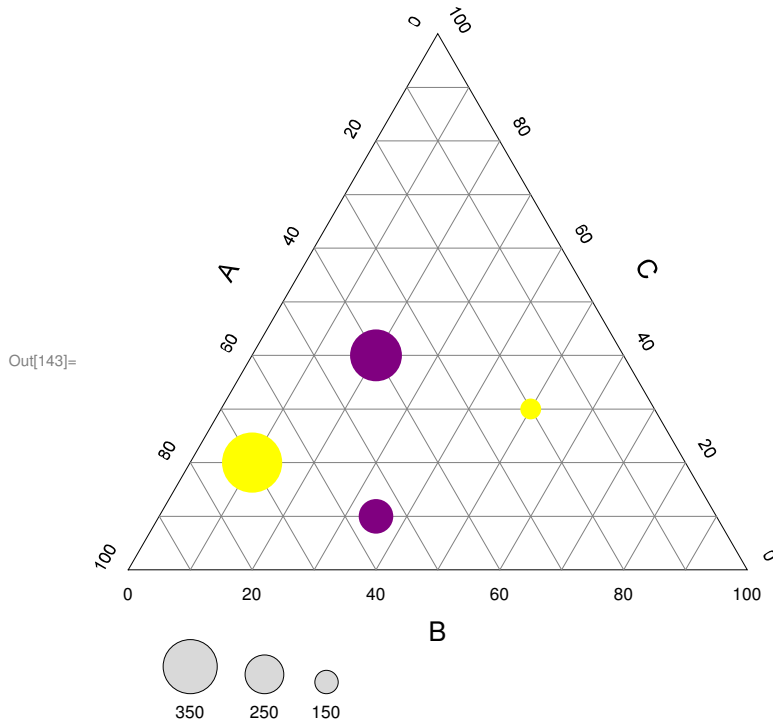


```
In[142]:= TrianglePlot[{{.2, .5, .3, 120}, {.7, .1, .2, 350},
  {.55, .35, .1, 200}, {.4, .2, .4, 300}}, {100, 400}, {"A", "B", "C"}, True]
```



Drobnou úpravou v syntaxi můžeme doplnit bodové znaky barvou - což by bylo možné využít např. pro odlišení kvalitativních znaků:

```
In[143]:= TrianglePlot[{{.2, .5, .3, 120}, {.7, .1, .2, 350}, Yellow},
  {{.55, .35, .1, 200}, {.4, .2, .4, 300}, Purple}},
  {100, 400}, {"A", "B", "C"}, True]
```



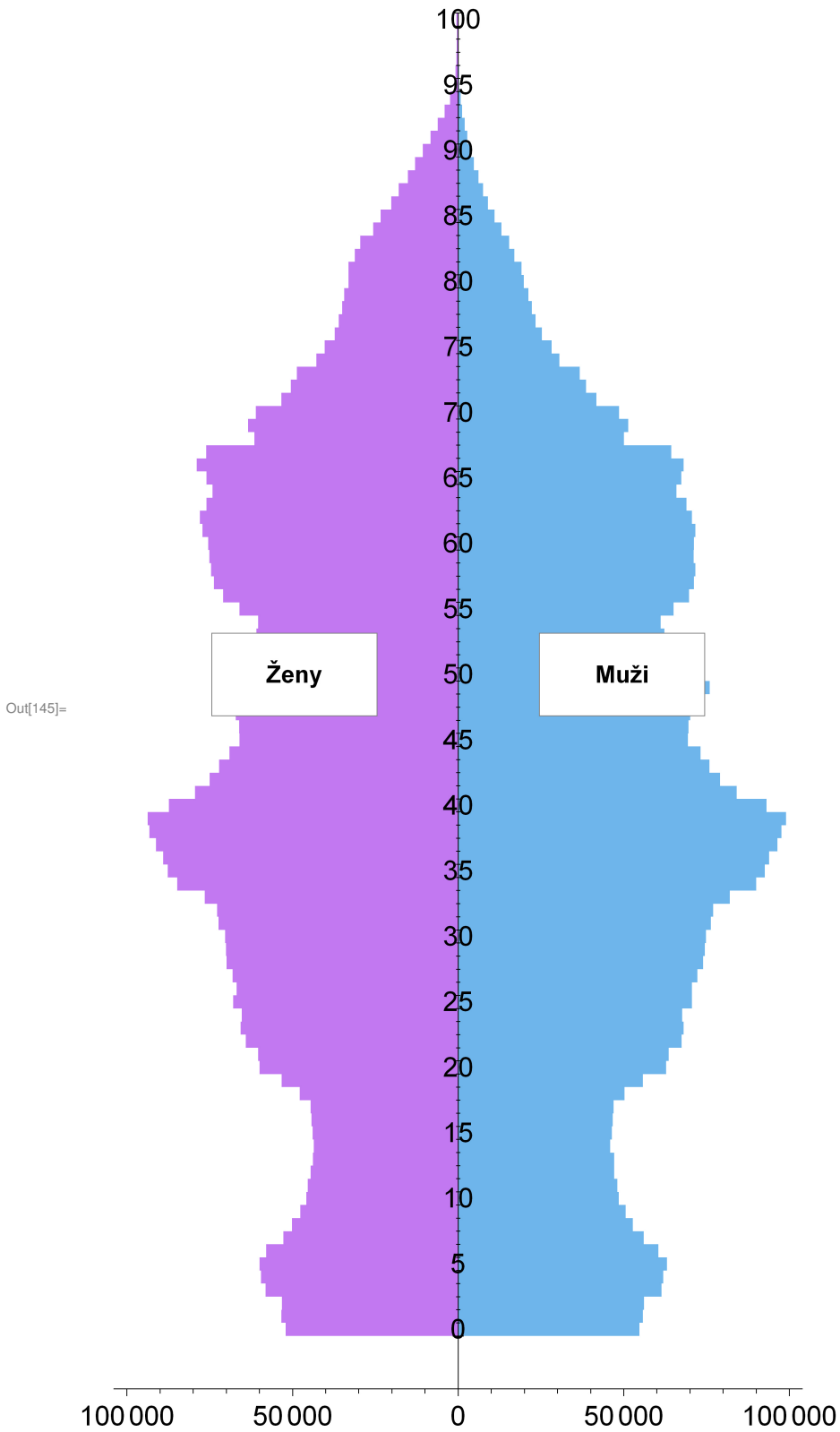
## 7.2 Věková pyramida

Věková pyramida je de facto druhem histogramu, resp. spojením dvou histogramů, z nichž jeden udává počet mužů podle věkových kategorií, druhý pak počet žen podle stejných kategorií. Jeho vytvoření v programu *Mathematica* si ukažme jako příklad složitějšího zadání grafu prezentujícího už poměrně značné množství číselných údajů. *Mathematica* umožňuje import datových souborů všech běžných formátů, v tomto případě šlo o soubor excelu (umístění souboru musí být definováno zcela přesně):

```
In[144]:= stat = Import[FileNameJoin[{NotebookDirectory[], "1300641410.xlsx"}], "Data"];
```

Instrukce ke zpracování dat pak mají tuto podobu:

```
In[145]:= PairedBarChart[Map[Round[Total[#]] &, stat[[1, 7 ;; 107, 6 ;; 9]]],
  Map[Round[Total[#]] &, stat[[1, 7 ;; 107, 2 ;; 5]]], ImageSize -> 440,
  ChartStyle -> {{ColorData["Pastel", 0], ColorData["Pastel", 1]}, None, None},
  BarSpacing -> {None, None, None}, ChartLabels -> {Placed[
    {Graphics[{FaceForm[White], EdgeForm[Gray], Rectangle[{0, 0}, {2, 1}],
      Text[Style["Ženy", Bold, 14], Scaled[ {.5, .5} ]]}, ImageSize -> 100],
    Graphics[{FaceForm[White], EdgeForm[Gray], Rectangle[{0, 0}, {2, 1}],
      Text[Style["Muži", Bold, 14], Scaled[ {.5, .5} ]]}, ImageSize -> 100]},
  Center], None, Placed[Table[If[Divisible[n, 5], n, ], {n, 0, 100}],
  Axis]}, AspectRatio -> 2, BaseStyle -> 16]
```



## 8 Práce s funkcemi

System *Mathematica* umožňuje provádět s funkcemi mj. symbolické derivování, integrování, výpočet limit, extrémů, výpočet součtů a součinů nekonečných řad, apod.

### 8.1 Derivace funkcí

O derivování funkce můžeme program *Mathematica* "požádat" dvěma způsoby:

- pomocí příkazu `D`,
- pomocí apostrofu.

Syntaxe obou možností je tato:

<code>D[funkce[x], x]</code>	$f'(x)$	derivace prvního řádu podle $x$
<code>D[funkce[x], {x, n}]</code>	$f''(x), f'''(x), f^{(4)}(x) \dots$	derivace $n$ - tého řádu podle $x$

Ukažme si obě možnosti na příkladu funkce  $y = x^3 - \cos(x)$ :

```
In[146]:= f[x_] := x^3 - Cos[x]
```

```
In[147]:= D[f[x], x]
```

```
Out[147]:= 3 x^2 + Sin[x]
```

```
In[148]:= D[f[x], {x, 2}]
```

```
Out[148]:= 6 x + Cos[x]
```

```
In[149]:= f'[x]
```

```
Out[149]:= 3 x^2 + Sin[x]
```

```
In[150]:= f''[x]
```

```
Out[150]:= 6 x + Cos[x]
```

Pokud není funkce určená k derivování příliš složitá, je nejjednodušší přímo ji vypsát do příkazu `D`:

```
In[151]:= D[x^2 - 2 x + 3, x]
```

```
Out[151]:= -2 + 2 x
```

Pokud chceme vypočítat derivaci v bodě, nejdříve funkci zderivujeme (výsledek bude výraz) a následně do výrazu dosadíme. Pro zopakování z kapitoly 3.2: dosazení se provádí pomocí příkazu `výraz /. x -> a`, kde se operátor `/.` označuje jako dosazovací operátor, znak `->` jako tzv. transformační pravidlo. Např. hodnota derivace funkce  $y = x^4 - 2x$  v bodě 3 je:

```
In[152]:= v = D[x^4 - 2 x + 1, x]
```

```
Out[152]:= -2 + 4 x^3
```

In[153]:=  $v / . x \rightarrow 3$ 

Out[153]:= 106

## 8.2 Integrál

Program *Mathematica* lze použít i pro výpočet neurčitých a určitých integrálů funkcí. Příslušný příkaz je `Integrate` a použitá syntaxe je následující:

<code>Integrate[f, x]</code>	neurčitý integrál $\int f dx$
<code>Integrate[f, {x, a, b}]</code>	určitý integrál $\int_a^b f dx$
<code>Integrate[f, {x, a, b}, {y, c, d}]</code>	dvojný určitý integrál $\int_a^b \int_c^d f dx dy$
<code>NIntegrate[f, {x, a, b}]</code>	numerická aproximace určitého integrálu $\int_a^b f dx$

Na tomto místě je třeba připomenout, že výpočet neurčitých integrálů je výrazně složitější, než derivování. Zatímco derivovat lze libovolnou - tedy i velmi složitou - funkci, u neurčitých integrálů mohou nastat 3 možnosti:

- primitivní funkci lze vyjádřit jako standardní funkce (exponenciální, logaritmické, trigonometrické, apod.),
- primitivní funkci lze vyjádřit jen pomocí speciálních funkcí,
- primitivní funkci není možné vyjádřit ani jako standardní, ani jako speciální funkce.

Podívejme se, jak se s jednotlivými možnostmi “vypořádá” *Mathematica*. Do první kategorie patří např. funkce  $y = x^2 - 3x + 2$ :

In[154]:= `Integrate[x^2 - 3 x + 2, x]`Out[154]=  $2 x - \frac{3 x^2}{2} + \frac{x^3}{3}$ 

Obdobně i u funkce  $y = \frac{1}{x^5+3}$  relativně snadno nalezne *Mathematica* primitivní funkci:

$$\text{In[155]:= Integrate}\left[\frac{1}{x^5 + 3}, x\right]$$

$$\text{Out[155]= } \frac{1}{20 \times 3^{4/5}} \left( 2 \sqrt{2(5 + \sqrt{5})} \operatorname{ArcTan}\left[\frac{-3 + 3\sqrt{5} + 4 \times 3^{4/5} x}{3 \sqrt{2(5 + \sqrt{5})}}\right] + \right. \\ \left. 2 \sqrt{10 - 2\sqrt{5}} \operatorname{ArcTan}\left[\frac{-3(1 + \sqrt{5}) + 4 \times 3^{4/5} x}{3 \sqrt{10 - 2\sqrt{5}}}\right] + \right. \\ \left. 4 \operatorname{Log}\left[3 + 3^{4/5} x\right] + (-1 + \sqrt{5}) \operatorname{Log}\left[3 + \frac{1}{2} 3^{4/5} (-1 + \sqrt{5}) x + 3^{3/5} x^2\right] - \right. \\ \left. (1 + \sqrt{5}) \operatorname{Log}\left[3 - \frac{1}{2} 3^{4/5} (1 + \sqrt{5}) x + 3^{3/5} x^2\right] \right)$$

Správnost integrace si můžeme ověřit zpětnou derivací (výsledný výraz ještě bude třeba zjednodušit na pokud možno co nejkratší tvar):

$$\text{In[156]:= } \mathbf{v = D[1 / (20 \times 3^{(4/5)}) (2 \operatorname{Sqrt}[2(5 + \operatorname{Sqrt}[5])] \operatorname{ArcTan}[-3 + 3 \operatorname{Sqrt}[5] + 4 \times 3^{(4/5)} x] / (3 \operatorname{Sqrt}[2(5 + \operatorname{Sqrt}[5])]) + 2 \operatorname{Sqrt}[10 - 2 \operatorname{Sqrt}[5]] \operatorname{ArcTan}[-3(1 + \operatorname{Sqrt}[5]) + 4 \times 3^{(4/5)} x] / (3 \operatorname{Sqrt}[10 - 2 \operatorname{Sqrt}[5]]) + 4 \operatorname{Log}[3 + 3^{(4/5)} x] + (-1 + \operatorname{Sqrt}[5]) \operatorname{Log}[3 + 1/2 \times 3^{(4/5)} (-1 + \operatorname{Sqrt}[5]) x + 3^{(3/5)} x^2] - (1 + \operatorname{Sqrt}[5]) \operatorname{Log}[3 - 1/2 \times 3^{(4/5)} (1 + \operatorname{Sqrt}[5]) x + 3^{(3/5)} x^2)], x]}$$

$$\text{Out[156]= } \frac{1}{20 \times 3^{4/5}} \left( \frac{4 \times 3^{4/5}}{3 + 3^{4/5} x} + \frac{(-1 + \sqrt{5}) \left(\frac{1}{2} 3^{4/5} (-1 + \sqrt{5}) + 2 \times 3^{3/5} x\right)}{3 + \frac{1}{2} 3^{4/5} (-1 + \sqrt{5}) x + 3^{3/5} x^2} - \right. \\ \left. \frac{(1 + \sqrt{5}) \left(-\frac{1}{2} 3^{4/5} (1 + \sqrt{5}) + 2 \times 3^{3/5} x\right)}{3 - \frac{1}{2} 3^{4/5} (1 + \sqrt{5}) x + 3^{3/5} x^2} + \right. \\ \left. \frac{8}{3^{1/5} \left(1 + \frac{(-3 + 3\sqrt{5} + 4 \times 3^{4/5} x)^2}{18(5 + \sqrt{5})}\right)} + \frac{8}{3^{1/5} \left(1 + \frac{(-3(1 + \sqrt{5}) + 4 \times 3^{4/5} x)^2}{9(10 - 2\sqrt{5})}\right)} \right)$$

$$\frac{1}{20 \times 3^{4/5}} \left( \frac{4 \times 3^{4/5}}{3 + 3^{4/5} x} + \frac{(-1 + \sqrt{5}) \left( \frac{1}{2} 3^{4/5} (-1 + \sqrt{5}) + 2 \times 3^{3/5} x \right)}{3 + \frac{1}{2} 3^{4/5} (-1 + \sqrt{5}) x + 3^{3/5} x^2} - \frac{(1 + \sqrt{5}) \left( -\frac{1}{2} 3^{4/5} (1 + \sqrt{5}) + 2 \times 3^{3/5} x \right)}{3 - \frac{1}{2} 3^{4/5} (1 + \sqrt{5}) x + 3^{3/5} x^2} + \frac{8}{3^{1/5} \left( 1 + \frac{(-3+3\sqrt{5}+4 \times 3^{4/5} x)^2}{18(5+\sqrt{5})} \right)} + \frac{8}{3^{1/5} \left( 1 + \frac{(-3(1+\sqrt{5})+4 \times 3^{4/5} x)^2}{9(10-2\sqrt{5})} \right)} \right)$$

In[157]:= **Simplify[v]**

Out[157]=  $\frac{1}{3 + x^5}$

Na druhou stranu např. integrál  $\int \frac{\ln(1-x)}{x} dx$  vyjádřit pomocí standardních funkcí nelze:

In[158]:= **Integrate[Log[1 - x] / x, x]**

Out[158]= -PolyLog[2, x]

Řešením je tedy speciální polylogaritmická funkce.

Příkladem neurčitého integrálu, který není řešitelný pomocí standardních ani speciálních funkcí, je např.  $\int \cos(\sin x) dx$ . *Mathematica* v tomto případě integraci neprovede, na výstupu jen zopakuje zadání:

In[159]:= **Integrate[Cos[Sin[x]], x]**

Out[159]=  $\int \text{Cos}[\text{Sin}[x]] dx$

I v případě, kdy není *Mathematica* schopna najít primitivní funkci, je nicméně možné přibližně numericky vyčíslit hodnotu určitého integrálu. Např. pro předchozí případ v mezích od 0 do 1:

In[160]:= **NIntegrate[Cos[Sin[x]], {x, 0, 1}]**

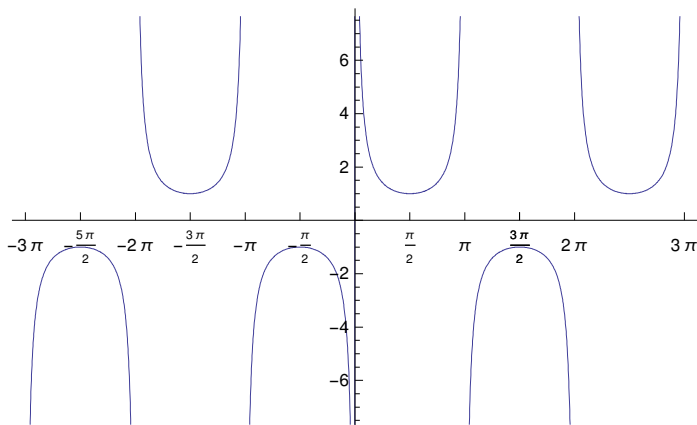
Out[160]= 0.86874

## 8.3 Limity

Limity se vypočítávají pomocí příkazu `Limit`:

<b>Limit[f[x], x → a]</b>	limita funkce f pro $x \rightarrow a$
<b>Limit[f[x], x → a, Direction → 1]</b>	limita funkce f pro $x \rightarrow a -$ (zleva)
<b>Limit[f[x], x → a, Direction → -1]</b>	limita funkce f pro $x \rightarrow a +$ (zprava)

Jako příklad limity zprava a zleva uvedeme funkci  $\frac{1}{\sin(x)}$  pro  $x \rightarrow \pi^-$  resp. pro  $x \rightarrow \pi^+$ . Funkce má tento graf:



```
In[161]:= Limit[1 / Sin[x], x -> Pi, Direction -> 1]
```

```
Out[161]= ∞
```

```
In[162]:= Limit[1 / Sin[x], x -> Pi, Direction -> -1]
```

```
Out[162]= -∞
```

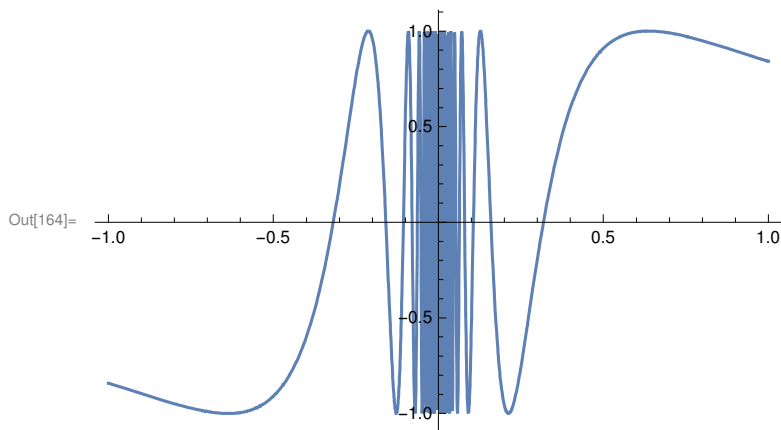
Na předchozím příkladu můžeme upozornit na jednu “zálužnost”: v případě, že limita zleva a limita zprava nejsou shodné, je zřejmé, že funkce v daném bodě limitu nemá - přesto nabízí vnitřní algoritmus programu *Mathematica* ve snaze nějaké řešení najít výsledek limity zprava:

```
In[163]:= Limit[1 / Sin[x], x -> Pi]
```

```
Out[163]= -∞
```

Podobně si snaží “poradit” i v jiném případě, kdy limita neexistuje: funkce  $\sin\left(\frac{1}{x}\right)$  osciluje pro  $x$  blížící se nule nekonečněkrát mezi  $+1$  a  $-1$ , limita pro  $x \rightarrow 0$  zjevně neexistuje:

```
In[164]:= Plot[Sin[1 / x], {x, -1, 1}]
```



System *Mathematica* vyjádří tuto skutečnost takto:



```
In[165]:= Limit[Sin[1/x], x -> 0]
```

```
Out[165]= Interval[{-1, 1}]
```

tj. určí, že jde o neznámou hodnotu, která ale leží někde uvnitř intervalu  $\langle -1, +1 \rangle$ . Podobně by skončila snaha najít limitu funkce  $\sin(x)$  pro  $x \rightarrow \infty$ :

```
In[166]:= Limit[Sin[x], x -> Infinity]
```

```
Out[166]= Interval[{-1, 1}]
```

Je tedy zřejmé, že i při použití programu *Mathematica* je pro správné určení limity vždy nezbytné znát průběh funkce.

## 9 Využití programu Mathematica ve statistice pro geografii

Nejpravděpodobnější využití programu *Mathematica* pro studenty geografie je její použití pro statistické analýzy, případně pro vizualizaci statistických dat. Proto se na tuto oblast zaměříme podrobněji a ukážeme si některé typické aplikace.

### 9.1 Třídění dat a rozdělení četností

Četností rozumíme počet prvků se stejnou hodnotou statistického znaku (každý statistický soubor tak generuje své rozdělení četností) nebo četností myslíme počet prvků s hodnotami znaku patřícími do určitého intervalu (nebo třídy) – pak se bavíme o tzv. skupinovém (intervalovém) rozdělení četností.

Absolutní četnost (označujeme  $n_i$ ) vyjadřuje absolutní hodnotou četnost zastoupených hodnot ve statistickém souboru, resp. v daném intervalu. Relativní četnost vyjadřuje četnost pomocí relativních hodnot, výpočet je dán vztahem  $f_i = \frac{n_i}{n}$ , tj. je dána podílem jednotlivých absolutních četností  $n_i$  k rozsahu souboru  $n$ . Může být uvedena desetinným číslem, nebo procentuálně. Kumulativní četnosti absolutní  $N_i$ , resp. relativní  $F_i$  udávají úhrnnou četnost statistických jednotek s hodnotami znaku menšími, nebo rovnými hodnotě znaku nebo horní hranici intervalu, při seřazení hodnot nebo intervalů podle pořadí neklesajících hodnot znaku. Kumulovanou četnost lze vyjadřovat a počítat z absolutních četností i z relativních četností.

### 9.2 Třídění dat do intervalů

Nejdříve si připomeňme používanou terminologii:

Hranice intervalu (neboli mez intervalu), at' už horní nebo dolní, určuje, které hodnoty do intervalu patří.

Délka intervalu (též rozpětí či šířka) je rozdíl (kladný) dvou po sobě následujících dolních (nebo horních) hranic intervalů.

Střed intervalu (označujeme ho  $x_s$ ) – je důležitou hodnotu, která při výpočtech z intervalového rozdělení četností zastupuje příslušný interval. Střed intervalu spočítáme jako aritmetický průměr horní a dolní hranice intervalu, neboli:  $(a+b) / 2$ , kde  $a$  ( $b$ ) je dolní (horní) hranice intervalu.

Typologie intervalů:

$<a; b>$  – uzavřený interval, množina všech  $x$ , pro která platí  $a \leq x \leq b$

$(a; b)$  – otevřený interval, množina všech  $x$ , pro která platí  $a < x < b$

$<a; b)$  – uzavřený interval zleva, množina všech  $x$ , pro která platí  $a \leq x < b$

$(a; b>$  – uzavřený interval zprava, množina všech  $x$ , pro která platí  $a < x \leq b$

Jednotlivé intervaly, do kterých sledovaný statistický soubor rozdělíme, vzniknou rozříděním jeho hodnot podle určitých kritérií:

- každý interval je přesně vymezen svojí horní a dolní hranicí,
- jsou vymezeny tak, aby šel každý prvek jednoznačně zařadit (intervaly se nesmí překrývat),
- šířka intervalů by měla být stejná (pro snadnější výpočty),
- počet intervalů volit optimálně („ani málo, ani příliš“).

Exaktní pravidla pro určení optimálního počtu intervalů neexistují, celý algoritmus bude vždy obsahovat subjektivní prvek. Přesto se setkáme s doporučeními, jak postupovat. Následující algoritmus představuje jedno z nich:

- a) určíme  $R$  jako rozdíl mezi maximální a minimální hodnotou (jedná se o variační rozpětí) sledovaného souboru, tzn.  $R = x_{\min} - x_{\max}$ ,
- b) výpočet počtu intervalů (tříd) označme  $k$  – rozdělíme ho na tři případy:
- je-li rozsah souboru  $n > 100$ , pak  $k = 10 \log n$ ,
  - je-li rozsah souboru  $40 < n \leq 100$ , pak  $k = 2\sqrt{n}$ ,
  - je-li rozsah souboru  $n \leq 40$ , pak  $k = 1 + 1.4426 \ln n$ ,
- c) výpočet šířky intervalu  $h$  je pak dán vztahem  $h = \frac{R}{k}$ .

Jak už bylo uvedeno výše, volba počtu intervalů se těmito pravidly nemusí řídit, může být intuitivní, provedená na základě analýzy struktury studovaných dat nebo na základě zkušeností.

### 9.3 Grafické vyjádření rozdělení četností

Zjištěné četnosti nejpřehledněji uvádíme v tabulkách intervalového (skupinového) rozdělení četností. Pro přehlednost, nadhled, nebo lepší orientaci prezentujeme data z těchto tabulek graficky, nejčastěji pomocí histogramu, polygonu a součtové čáry.

Histogram je graf vyjadřující rozložení četností ve statistickém souboru. Jedná se o graf sloupcový, při jeho konstrukci nezáleží na tom, zda jako zdrojová data uvažujeme absolutní, nebo relativní četnosti (pro oba způsoby vypadá diagram stejně). Na vodorovnou osu  $x$  nanášíme intervaly v příslušných jednotkách, na ose svislé  $y$  se vynáší absolutní (relativní) četnosti. Jak již bylo řečeno, jedná se o sloupcový graf, kde šířka sloupce odpovídá délka (šířka) intervalu a výšce pak četnost v daném intervalu. Z vhodně sestrojeného histogramu lze vypočítat rozložení hodnot ve statistickém souboru, jejich rozmístění okolo střední hodnoty, rovněž jejich rozptyl v souboru a dají se určit další charakteristiky, jako například modální interval aj.

Polygon je obdobou histogramu, také vyjadřuje rozložení četností ve statistickém souboru, liší se pouze typem grafu. Zatímco v případě histogramu se jedná o sloupcový graf, u polygonu jde o spojnicový typ grafu. Z vlastností polygonu vyplývá, že v případě jeho sestavení z relativních četností ohraničuje křivka polygonu plochu o velikosti 1 (v případě vyjádření v procentech pak 100 %). Polygon i histogram tak znázorňují stejné údaje poněkud odlišným způsobem, nezáleží na výběru z absolutních nebo relativních četností.

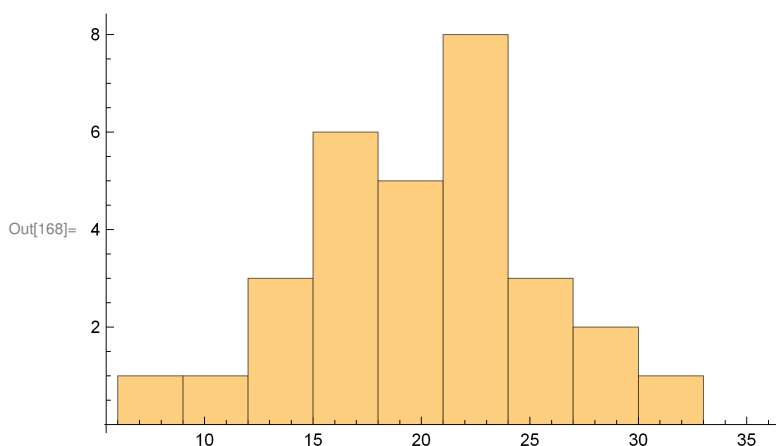
Součtová čára slouží pro znázornění kumulovaných četností. Při konstrukci se vynáší hodnoty kumulovaných četností (nezáleží na tom, zda absolutní, či relativní, ale častěji se používají relativní) k horním hranicím intervalů, body se spojí lomenou čarou. Z grafu součtové čáry lze vyčíst řadu charakteristik, mj. hodnoty kvartilů, medián atd.

## Tvorba histogramu:

Tvorba histogramu je v programu *Mathematica* poměrně jednoduchá, používá se příkaz `Histogram[data, { $x_{\min}$ ,  $x_{\max}$ ,  $d$ }]`, kde `data` je množina hodnot, pro kterou vytváříme histogram,  $x_{\min}$  a  $x_{\max}$  jsou meze hodnot, pro které z dat vytváříme histogram a konečně  $d$  je šířka intervalu v histogramu. Na obrázcích ukážeme, jak změnit podobu histogramu právě volba šířky intervalu:

```
In[167]:= data = {22, 25, 19, 17, 25, 31, 8, 17, 18, 15, 27, 17, 16,
             22, 24, 21, 18, 23, 18, 13, 21, 12, 22, 10, 16, 13, 29, 19, 21, 22};
```

```
In[168]:= Histogram[data, {6, 38, 3}]
```



## 9.4 Základní statistické charakteristiky : charakteristiky úrovně a polohy

Statistickými charakteristikami (ukazateli) úrovně, resp. polohy statistického souboru, rozumíme hodnoty zkoumaného znaku, které udávají velikost jevu v daném souboru a udávají polohu četností. Slouží k porovnávání dvou i více souborů, charakteristiky úrovně vlastně zastupují všechny hodnoty statistického souboru (typicky např. aritmetický průměr). Nejčastěji používanými charakteristikami úrovně jsou střední hodnoty (průměry, modus, medián apod.), dále sem řadíme např. kvantily (kvartily, decily, percentily).

Střední hodnoty patří k nejdůležitějším a nejpoužívanějším charakteristikám statistických souborů vůbec. Obzvláště průměr, modus a medián. O středních hodnotách se bavíme v případě různých druhů průměrů (aritmetický, harmonický, geometrický, vážený), řadíme sem také modus, medián a aritmetický střed.

Patrně nejpoužívanější statistickou charakteristikou je aritmetický průměr. Jeho výpočet je navíc velmi jednoduchý – jde o úhrn hodnot statistického znaku, dělený rozsahem souboru. Mezi základní vlastnosti aritmetického průměru patří:

- algebraický součet všech odchylek jednotlivých hodnot znaku od aritmetického průměru je roven nule,
- je-li znak konstantní, průměr je roven této konstantě,
- přičteme-li ke všem hodnotám znaku konstantu  $k$ , zvětší se i průměr o tuto konstantu,
- vynásobíme-li všechny hodnoty znaku konstantou  $k$ , je i průměr  $k$ -krát větší.

Kromě té výhody, že výpočet aritmetického průměru je velmi jednoduchý, má tato charakteristika i některé nevýhody, a to zejména tu, že nemusí vždy podávat správnou informaci. Může být zkreslen extrémní (výraznou maximální nebo minimální) hodnotou v případě, že vycházíme ze souboru s nižším rozsahem, rovněž rozdělení hodnot v souboru může mít dva nebo více vrcholů, a ty jedním ukazatelem nelze popsat. Pak mluvíme o „typickém“ průměru – kdy je většina hodnot souboru „blízká“ průměru – a naopak o „netypickém“ průměru.

## Výpočet aritmetického průměru:

K výpočtu aritmetického průměru slouží funkce `Mean[{x1, x2, x3, ...}]`. Pokud ji aplikujeme na data z předchozí podkapitoly (jen na připomenutí: “//N” přidáváme k příkazu, protože chceme přibližný, nikoliv přesný výsledek), dostáváme:

```
In[169]:= Mean[data] // N
```

```
Out[169]= 19.3667
```

Vážený průměr se využívá v případě, kdy prvky statistického souboru mají různou důležitost, tj. že každému prvku statistického souboru  $x_i$  je přiřazena jeho váha  $n_i$ . Typickým, i když negeografickým, příkladem jsou získané známky ze zkoušek absolvovaných předmětů, váhami pak jsou kredity příslušné těmto předmětům. Vážený průměr dostaneme jako součet součinů prvků a jejich vah dělený celkovým součtem vah. Pro výpočet aritmetického průměru rozříděného statistického souboru (kdy neznáme vstupní data), se používá právě váženého průměru, jde pak o průměr středů intervalů a jednotlivé váhy jsou pak četnosti příslušné jednotlivým intervalům.

## Výpočet váženého průměru:

Vážený průměr nemá v programu *Mathematica* samostatnou funkci, snadno ji ale můžeme sami nadefinovat ( $x_i$  je hodnota  $i$ -tého znaku,  $n_i$  jeho váha):

```
In[170]:= VazAritmPrum[x_List, n_List] :=  
Sum[x[[i]] * n[[i]], {i, 1, Length[x]}] / Sum[n[[i]], {i, 1, Length[x]}]
```

Pokud by např. v souboru znaků {1,2,3,4,5,6,7,8} byla prvnímu znaku přiřazena váha 8, druhému 12, třetímu 18 ...), vypadal by výpočet takto:

```
In[171]:= VazAritmPrum[{1, 2, 3, 4, 5, 6, 7, 8}, {8, 12, 18, 36, 63, 46, 23, 14}] // N
```

```
Out[171]= 4.97273
```

Geometrický průměr se používá v případech, kdy hodnoty tvoří alespoň přibližně geometrickou řadu. Tehdy má smysl uvažovat o použití geometrického průměru. V geografii se pomocí geometrického průměru analyzují zpravidla časové řady, typickou úlohou je výpočet průměrného tempa růstu. Geometrický průměr se počítá jako  $n$ -tá odmocnina ze součinu všech hodnot souboru:

## Výpočet geometrického průměru:

Rovněž geometrický průměr je možné snadno nadefinovat:

```
In[172]:= GeomPrum[x_List] := N[(Apply[Times, x]) ^ (1 / Length[x])]
```

Geometrický průměr dat uvedených na počátku kapitoly u histogramu by byl:

```
In[173]:= GeomPrum[data]
```

```
Out[173]= 18.5758
```

Geometrický průměr z čísel 3, 4 a 5:

```
In[174]:= GeomPrum[{3, 4, 5}]
```

```
Out[174]= 3.91487
```

Aritmetický střed je spíše doplňkový ukazatel, popřípadě podává prvotní informaci o rozložení hodnot ve statistickém souboru. V případě, že jsou hodnoty v něm rozloženy rovnoměrně, podává poměrně kvalitní informaci v tom smyslu, že se aritmetický střed v takovém případě blíží aritmetickému průměru. Z jeho vlastní definice (jedná se o aritmetický průměr maximální a minimální hodnoty v souboru) pak plynou i případné

nevýhody. Je-li maximální, nebo minimální hodnota výrazně „vychýlena“ či „vzdálena“ od ostatních hodnot, není jeho použití vhodné a nemá příliš velkou vypovídající hodnotu.

## Výpočet aritmetického středu:

Pro výpočet aritmetického středu můžeme použít např. funkce `Min[x1, x2, x3, ...]`, která vybírá ze souboru hodnot aritmeticky nejnižší, a funkci `Max[x1, x2, x3, ...]`:

```
In[175]:= 
$$\frac{\text{Min}[\text{data}] + \text{Max}[\text{data}]}{2} // N$$

```

```
Out[175]= 19.5
```

Modus je nejčetnější (nejčastější) hodnota kvantitativního znaku studovaného souboru, to v případě, že vycházíme z neroztříděného souboru, tedy ze všech jeho hodnot. Na první pohled je tak zřejmé, že pro snadné nalezení modu je vhodné seřadit hodnoty znaku vzestupně nebo sestupně. Důležitost modu se projeví při vystižení typické hodnoty znaku v daném souboru a následně při porovnávání typických hodnot souborů.

## Výpočet modu:

Pro výpočet modu se užívá funkce `Commonest[list]`:

```
In[176]:= Commonest [data]
```

```
Out[176]= {22}
```

Medián je prvek řady (hodnot sledovaného znaku), uspořádané v neklesajícím (rostoucím) pořadí, který ji rozděluje na dvě části v tom smyslu, že polovina prvků této řady má menší hodnotu znaku a polovina má větší hodnotu znaku, než je hodnota mediánu. Jinými slovy lze prohlásit, že za medián považujeme hodnotu, která nám dělí vzestupně seřazené hodnoty statistického souboru na dvě stejné poloviny. Výhodou mediánu je, že zachycuje úroveň (polohu) hodnot lépe než průměr.

## Výpočet mediánu:

Pro výpočet mediánu se užívá funkce `Median[list]`:

```
In[177]:= Median [data]
```

```
Out[177]= 19
```

Kvantily se řadí mezi charakteristiky úrovně, střední hodnotou je však pouze jeden z nich, a to medián. Kvantily obecně fungují na stejném principu jako medián. Jak již bylo uvedeno, za medián považujeme hodnotu, která dělí vzestupně seřazené hodnoty statistického souboru na dvě stejné poloviny. Kvartily jsou takové hodnoty, které od sebe oddělují čtvrtiny vzestupně seřazených hodnot souboru. Jsou tedy celkem tři. První (dolní) kvartil odděluje první čtvrtinu hodnot od zbylých tří čtvrtin, druhý (prostřední) kvartil odděluje první dvě čtvrtiny od zbylých dvou a je tedy totožný s mediánem, třetí (horní) kvartil odděluje první tři čtvrtiny hodnot od poslední čtvrtiny. Obdobně v souboru identifikujeme decily, kterých je v každém statistickém souboru celkem devět a dělí ho na jednotlivé desetiny, a konečně percentily, které ho dělí na setiny. Percentilů je v souboru 99.

## 9.5 Základní statistické charakteristiky : charakteristiky variability

Charakteristiky variability jsou hodnoty, které charakterizují stupeň proměnlivosti statistického znaku (resp. hodnot sledovaného jevu) v daném statistickém souboru. Měříme proměnlivost vzhledem k typické hodnotě souboru, zpravidla vzhledem k průměru nebo mediánu. Charakteristiky variability jsou důležitým doplňkem informací, které poskytují střední hodnoty.

Nejjednodušší ukazatel variability souboru je variační rozpětí, určí se jako rozdíl minimální a maximální hodnoty ve sledovaném souboru.

Jedná se o ukazatel jednoduchý, ale protože závisí pouze na dvou extrémních hodnotách, nemusí být dostatečně výstižný, maximální a minimální hodnota může být „nahodilá“. Tato ne příliš dokonalá míra variability slouží především k první informaci o variabilitě souboru.

### Výpočet variačního rozpětí:

Pro výpočet variačního rozpětí můžeme použít vztah:

```
In[178]:= R[xMin_, xMax_] := xMax - xMin
```

Průměrné odchylky vyjadřují míru odlišnosti (variance) od střední hodnoty (průměru, mediánu). Jsou doplňkovou informací ke střední hodnotě a spočítají se jako aritmetický průměr absolutních hodnot odchylek (rozdílů) všech hodnot znaku od střední hodnoty (aritmetického průměru, mediánu...). Pokud vydělíme průměrnou odchylku střední hodnotou (průměrem, nebo mediánem), dostaneme relativní bezrozměrnou míru.

### Výpočet průměrné odchylky:

Pro výpočet průměrné odchylky můžeme použít vztah MeanDeviation[list]:

```
In[179]:= MeanDeviation[data] // N
```

```
Out[179]= 4.25778
```

Střední diference je definována jako aritmetický průměr absolutních hodnot všech možných vzájemných rozdílů n jednotlivých hodnot sledovaného znaku x. Je to vhodná míra variability pro soubory s malým rozsahem, v ostatních případech je její výpočet zbytečně pracný.

### Výpočet střední diference:

Výpočet střední diference podle definičního vztahu:

```
In[180]:= StredDif[data_List] := Sum[  
  Abs[data[[i]] - data[[j]]  
  / (Length[data] * (Length[data] - 1)),  
  {i, 1, Length[data]}, {j, 1, Length[data]}] // N
```

A aplikace na náš pokusný soubor dat:

```
In[181]:= StredDif[data]
```

```
Out[181]= 6.14483
```

Rozptyl vypočítáme jako průměr ze čtverců odchylek jednotlivých hodnot znaku od jejich aritmetického průměru. Vybrané vlastnosti rozptylu: pokud odečteme od všech hodnot statistického souboru stejnou konstan-

tou  $k$ , rozptyl souboru zůstane nezměněn, po vynásobení všech hodnot statistického souboru stejnou konstantou  $k$  musíme rozptyl vynásobit druhou mocninou této konstanty.

## Výpočet rozptylu:

Výpočet rozptylu umožňuje funkce `Variance[list]`:

```
In[182]:= Variance[data] // N
```

```
Out[182]:= 28.8609
```

V praxi se častěji než s rozptylem setkáváme se směrodatnou odchylkou, která je definována jako druhá odmocnina z rozptylu a vlastně se jedná o míru rozptylu hodnot sledovaného znaku  $x_i$  kolem průměru.

Vybrané vlastnosti směrodatné odchylky: pokud odečteme od všech hodnot statistického souboru stejnou konstantu  $k$ , směrodatná odchylka zkoumaného souboru zůstane nezměněna; po vynásobení všech hodnot statistického souboru konstantou  $k$ , se směrodatná odchylka musí vynásobit také touto konstantou.

## Výpočet směrodatné odchylky:

Výpočet rozptylu je vzhledem k definici velmi jednoduchý, jde o:

```
In[183]:= SmerodOdch[x_List] := Sqrt[Variance[x]]
```

V našem modelovém případě:

```
In[184]:= SmerodOdch[data] // N
```

```
Out[184]:= 5.37224
```

Variační koeficient je dán poměrem směrodatné odchylky a aritmetického průměru a z definice tohoto poměru plyne, že jde o ukazatel (míru) bezrozměrný.

## Výpočet variačního koeficientu:

Variační koeficient je možné vypočítat jako podíl směrodatné odchylky a aritmetického průměru, tj.

```
In[185]:= VarKoeff[x_List] := Sqrt[Variance[x]] / Mean[x]
```

V našem modelovém případě:

```
In[186]:= VarKoeff[data] // N
```

```
Out[186]:= 0.277396
```

### 9.6 Základní statistické charakteristiky : charakteristiky šikmosti

Charakteristikami šikmosti (symetrie, asymetrie) myslíme míry (čísla), která charakterizují nerovnoměrné (nesouměrné) rozložení četností ve statistickém souboru. Pomocí nich jsme schopni odhadnout tvar rozdělení četností (resp. jeho souměrnost, nebo nesouměrnost). Souměrné rozdělení četností má míry šikmosti nulové.

Míra šikmosti je jednoduchou charakteristikou šikmosti co do výpočtu, ale jinak je to míra poměrně nedokonalá, ovlivněná maximální a minimální hodnotou souboru, které mohou být „nahodilé“. Hodnoty míry šikmosti se pohybují v intervalu  $(-1;1)$ .



Koeficient šikmosti je, na rozdíl od míry šikmosti založené na variačním rozpětí, nebo založené na rozpětí kvantilů, dokonalejším ukazatelem. Je definován jako aritmetický průměr z třetích mocnin odchylek jednotlivých hodnot znaku od aritmetického průměru vydělený třetí mocninou směrodatné odchylky. Je-li koeficient  $> 0$ , pak je rozdělení četností zešikmeno doleva (kladná šikmost), pokud je roven 0, pak je rozdělení četností souměrné (nulová šikmost) a je-li koeficient  $< 0$ , pak je rozdělení četností zešikmeno doprava (záporná šikmost).

## Výpočet koeficientu šikmosti:

Koeficient šikmosti je v programu *Mathematica* předdefinován jako `Skewness[list]`. V našem případě:

```
In[187]:= Skewness[data] // N
```

```
Out[187]= 0.0205523
```

### 9.7 Základní statistické charakteristiky : charakteristiky špičatosti

Jedná se o čísla, která charakterizují koncentraci prvků souboru v blízkosti určité hodnoty znaku, jejich úkolem je poskytnout představu o tvaru rozdělení četností co do špičatosti nebo plochosti.

Koeficient špičatosti je definován jako průměrná hodnota součtu čtvrtých mocnin odchylek hodnot znaku od aritmetického průměru dělených čtvrtou mocninou směrodatné odchylky. Pokud je koeficient  $> 0$ , pak je rozdělení četností kladně zašpičatělé (špičaté), je-li nulový, pak je rozdělení četností normálně zašpičatělé a pokud je  $< 0$ , pak je rozdělení četností záporně zašpičatělé (ploché).

## Výpočet koeficientu špičatosti:

Koeficient špičatosti je v programu *Mathematica* předdefinován jako `Kurtosis[list]`. V našem případě:

```
In[188]:= Kurtosis[data] // N
```

```
Out[188]= 2.76363
```

### 9.8 Závislosti mezi náhodnými veličinami

V praxi se často dostáváme do situace, kdy je nutné analyzovat a charakterizovat vztah dvou jevů (resp. dvou náhodných veličin), tento vztah (případně závislost) změřit, a pokud existuje, tak ho vyjádřit matematicky, nejlépe pomocí funkce. Nejčastěji se to dělá pomocí korelačního počtu, resp. regresní analýzou.

## Korelační počet

Úkolem korelačního počtu je změřit těsnost vztahu mezi dvěma proměnnými, nebo těsnost změny hodnoty znaku závisle proměnné při změně hodnoty znaku nezávisle proměnné. Stanovení této těsnosti (těsnosti korelační závislosti) je nutným krokem, jež předchází regresní analýze a vyjádření této závislosti matematickou funkcí. Korelační koeficient se řadí k nejdůležitějším charakteristikám hodnocení korelační závislosti. Předpokládá linearitu studovaných proměnných. Zmíněnou těsnost závislosti dvou jevů (dvou náhodných veličin)  $X$  a

Y změříme pomocí charakteristiky „koeficient korelace“:

$$r_{xy} = \frac{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\frac{1}{n} \sum_{i=1}^n (x_i - \bar{x})^2 \cdot \frac{1}{n} \sum_{i=1}^n (y_i - \bar{y})^2}}$$

Použití korelačního koeficientu předpokládá normální rozdělení obou výběrů (pokud tomu tak není, je třeba oba výběry na toto rozdělení převést), další podmínkou je linearita vztahu jevů X a Y, tzn. že regresní funkcí musí být přímka. Výše zmiňovaný koeficient se nazývá v odborné literatuře často též „Pearsonův korelační koeficient“. V praxi se též můžeme setkat ještě s tzv. „Spearmanovým koeficientem“, který nebere v potaz jednotlivé hodnoty sledovaných jevů, ale jejich pořadí. Důležitým prvkem korelační a regresní analýzy, který nám může okamžitě napovědět o vztahu mezi dvěma veličinami je tzv. „korelační pole (diagram)“, což je bodový graf zobrazující obě náhodné veličiny X a Y.

Pro (Pearsonův) korelační koeficient platí, že:

- hodnoty se pohybují v intervalu  $\langle -1; 1 \rangle$ ,
- v případě, že  $r_{xy} = 1$ , hovoříme o tzv. přímé korelační závislosti, kdy přírůstek nezávisle proměnné znamená přírůstek závisle proměnné,
- v případě, že  $r_{xy} = -1$ , hovoříme o tzv. nepřímé korelační závislosti, kdy přírůstek nezávisle proměnné znamená úbytek závisle proměnné,
- statistická závislost (resp. její významnost) se posuzuje pomocí t-testu, testujeme korelační koeficient, testové kritérium t je dáno vztahem (t-rozdělení s  $n-2$  stupni volnosti):

$$t = \frac{r_{xy}}{\sqrt{1-r_{xy}^2}} \cdot \sqrt{n-2}$$

V programu *Mathematica* se korelační koeficient vypočte pomocí příkazu `Correlation[v1, v2]`, kde v1 a v2 jsou 2 vektory, resp. sady dat, např.  $\{a_1, a_2, \dots\}$  a  $\{b_1, b_2, \dots\}$ .

Jako příklad výpočtu se pokusíme prověřit těsnost korelační závislosti mezi proměnnými X a Y v tabulce:

x	25	45	34	192	136	218	221	201	228	158	64	75
y	155	930	383	1443	1069	1460	1208	1325	491	785	186	222

```
In[189]:= Correlation[{25, 45, 34, 192, 136, 218, 221, 201, 228, 158, 64, 75},
  {155, 930, 383, 1443, 1069, 1460, 1208, 1325, 491, 785, 186, 222.}]
```

```
Out[189]= 0.707573
```

## Regresní analýza

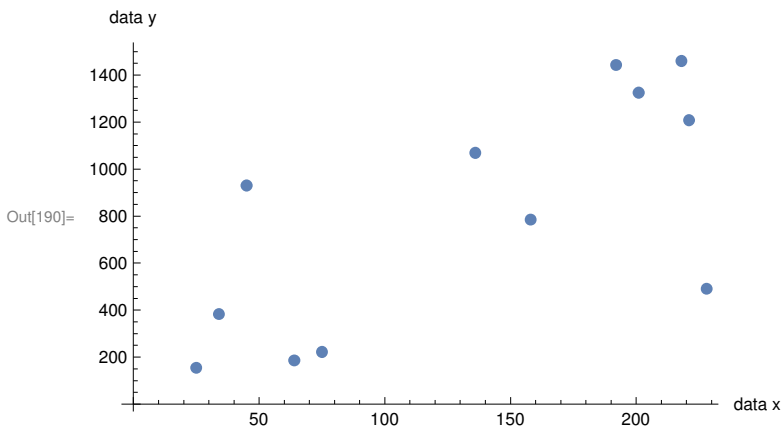
Cílem regresní analýzy je nalézt tvar (předpis) funkce, která co nejlépe vystihuje závislost mezi jevy X a Y (tzv. regresní funkce). Obvykle jej volíme tak, aby co nejvíce odpovídal vyšetřované nebo uvažované závislosti. Bývá zvykem volit regresní funkci s co nejmenším počtem regresních koeficientů, avšak dostatečně flexibilní a s požadovanými vlastnostmi (např. monotonie, předepsané hodnoty, asymptoty aj.). Vychází se přitom povětšinou ze zkušenosti, avšak v současné době se při realizaci regresní analýzy s využitím statistických softwarů dají často úspěšně použít vhodné databáze regresních funkcí. Regresní funkce rozdělujeme na lineární a nelineární (vzhledem k regresním koeficientům). Některé nelineární regresní funkce (např. kvadratickou, logaritmickou, exponenciální regresi) můžeme vhodnou transformací převést na lineární (např. mocnin-

nou nebo exponenciální funkci logaritmujeme). Jde sice o běžně používaný postup, kdy však nakonec řešíme jiný regresní model nežli původně uvažovaný.

Nejjednodušší případ regresní funkce je lineární regrese, kdy regresní čarou je přímka. Tato přímka je dána vztahem  $y = a + bx$ , který vyjadřuje výskyt hodnot  $y$  (závisle proměnná), očekávaných s největší pravděpodobností a podmíněných změnami  $x$  (nezávisle proměnná). Průběh regresní přímky (vyjádření koeficientů  $a$ ,  $b$ ) je nejčastěji počítán tzv. metodou nejmenších čtverců. Metoda spočívá v podmínce, aby se hledaná přímka co nejvíce přimykala bodům korelačního pole tak, že součet druhých mocnin (čtverců) vzdáleností bodů pole od přímky musí být minimální.

Příklad: pro proměnné  $x$  a  $y$  z předchozího příkladu vytvoříme nejdříve bodový graf:

```
In[190]:= ListPlot[{{25, 155}, {45, 930}, {34, 383}, {192, 1443}, {136, 1069}, {218, 1460},
  {221, 1208}, {201, 1325}, {228, 491}, {158, 785}, {64, 186}, {75, 222}},
  PlotStyle -> PointSize[0.02], AxesLabel -> {"data x", "data y"}]
```



```
In[191]:=
```

K výpočtu regresní přímky používá *Mathematica* příkaz `LinearModelFit`. Nejdříve si označíme soubor dat jako např. `data123`:

```
In[192]:= data123 = {{25, 155}, {45, 930}, {34, 383}, {192, 1443}, {136, 1069}, {218, 1460},
  {221, 1208}, {201, 1325}, {228, 491}, {158, 785}, {64, 186}, {75, 222}};
```

Regresní přímku pak získáme:

```
In[193]:= primka = LinearModelFit[data123, x, x]
```

```
Out[193]:= FittedModel[211.943 + 4.4544 x]
```

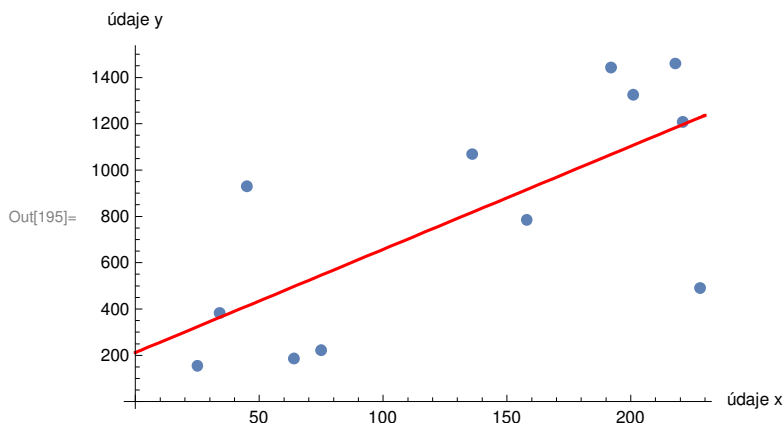
Pro převedení do tvaru funkce:

```
In[194]:= Normal[primka]
```

```
Out[194]:= 211.943 + 4.4544 x
```

Poměrně jednoduché je i její zobrazení v grafu:

```
In[195]:= Show[ListPlot[data123, PlotStyle -> PointSize[0.02]],
  Plot[primka[x], {x, 0, 230}, PlotStyle -> RGBColor[1, 0, 0]],
  AxesLabel -> {"údaje x", "údaje y"}]
```

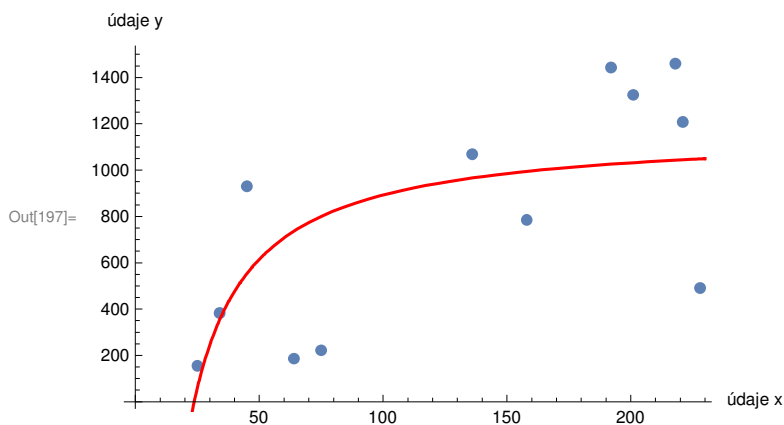


Podle stejného principu lze najít i nelineární regrese, příslušný příkaz je `NonlinearModelFit`  $\{ \{y_1, y_2, \dots\}, form, \{ \beta_1, \dots \}, x \}$ , kde je  $\{y_1, y_2, \dots\}$  soubor dat,  $form$  je předpis regresní funkce,  $\beta_1 \dots$  pak její parametry. Např. pokud bychom chtěli najít "regresní" hyperbolu:

```
In[196]:= krivka = NonlinearModelFit[data123, e / x + h, {e, h}, x]
```

Out[196]= FittedModel  $\left[ 1170.08 - \frac{27754.4}{x} \right]$

```
In[197]:= Show[ListPlot[data123, PlotStyle -> PointSize[0.02]],
  Plot[krivka[x], {x, 0, 230}, PlotStyle -> RGBColor[1, 0, 0]],
  AxesLabel -> {"údaje x", "údaje y"}]
```



## 10 Export a úprava souborů formátu shp v programu Mathematica

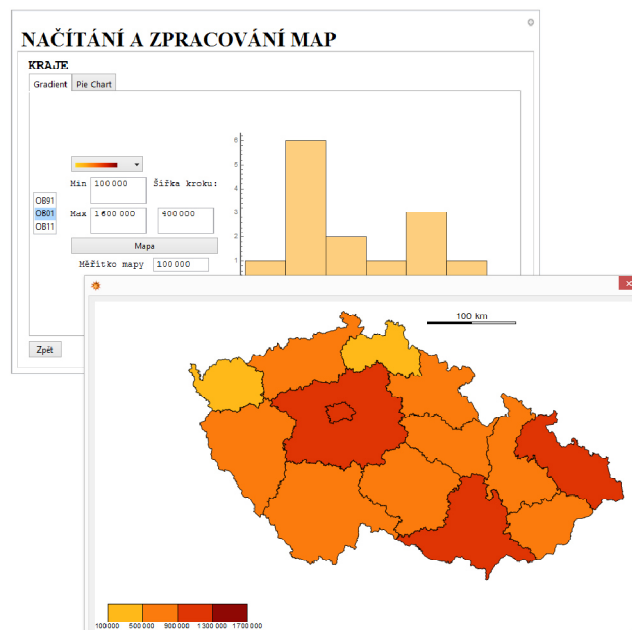
*Pozn. Pro přílišnou délku kódu u obou programů v této kapitole je v tištěné verzi schován. V internetové verzi (soubor \*.nb) je možné si ho zobrazit.*

Program *Mathematica* umožňuje i zpracování souborů shp. Jejich export a základní zpracování je poměrně komplikovaným úkolem, který čtenářům tohoto textu zjednodušíme programem:

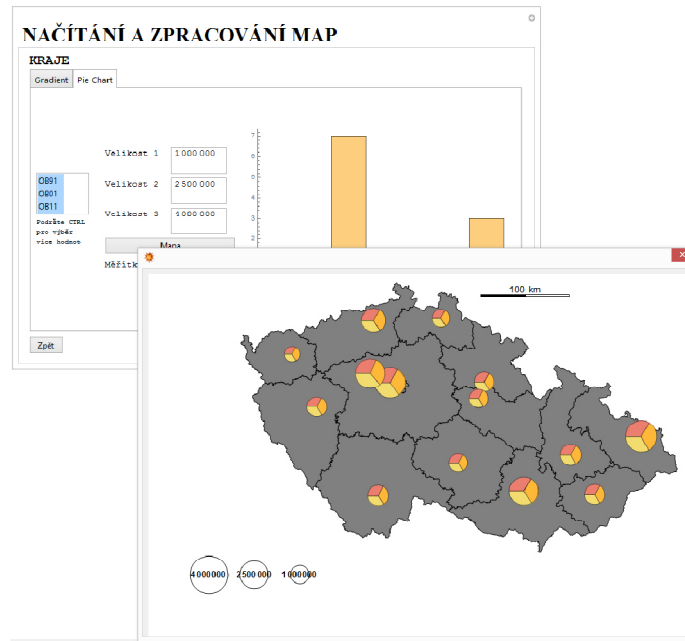


### Popis funkcí programu:

V 1. okně se zmáčkne tlačítko "Browse..." -> Musí se najít \*.shp soubor, který chceme načíst. Při úspěšném načtení se u "Cesta:" zobrazí umístění daného souboru. Stiskem "Načíst" se obsah \*.shp souboru načte a otevře se následující okno. V tomto okně je nutno zaškrtnout data, se kterými se má v dalším pracovat, výběr se potvrdí tlačítkem "Pokračuj", otevře se finální okno. V průběhu se dá vracet mezi jednotlivými okny pomocí tlačítka "Zpět". Ve finálním okně se provede výběr typu mapy (kartogram nebo koláčové diagramy). Histogram vpravo slouží jako přehled o tom, jaká data tam jsou zpracovávána a jakých dosahují hodnot. Jakmile si nastaví požadované parametry, mapa se vykreslí stiskem tlačítka "Mapa".



Obr. 1 : Mapa krajů ČR – počet obyvatel v roce 2001 (Kartogram)



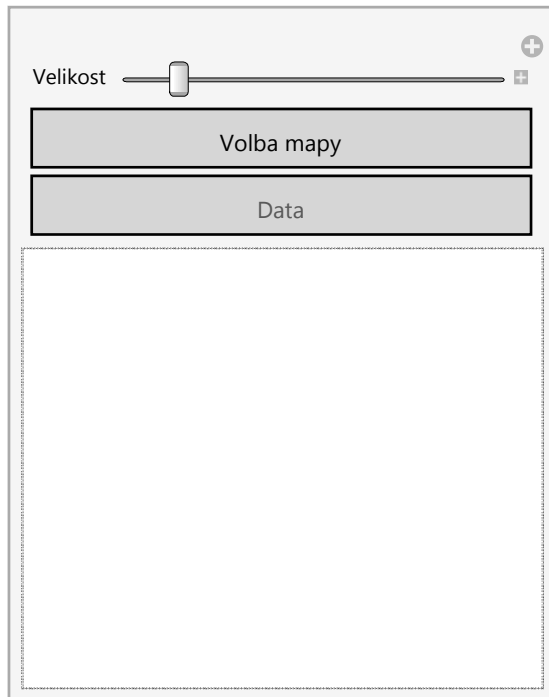
Obr. 2: Mapa krajů ČR – počet obyvatel v letech 1991, 2001 a 2011 (Koláčový diagram)

Následující program je určen pro zobrazení map jednotlivých krajů či okresů v České Republice. Před jeho spuštěním je třeba načíst potřebné soubory, obsahující data a mapy nutná ke správnému chodu programu.

```

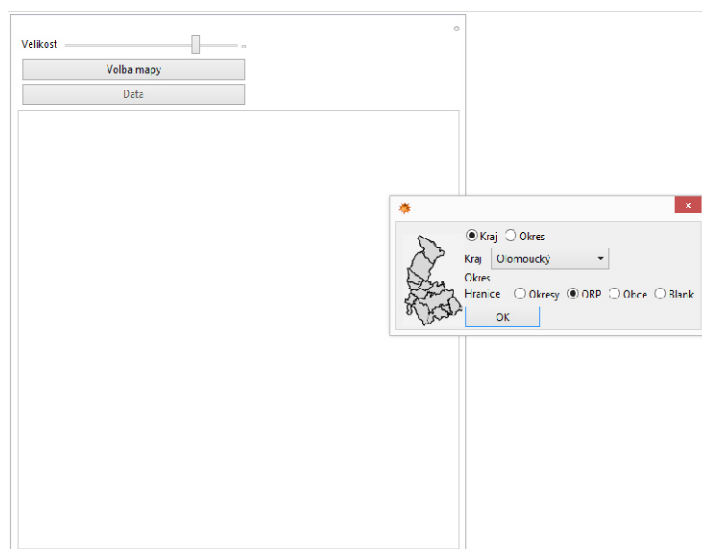
kraM =
  Flatten["Geometry" /. Import[NotebookDirectory[] <> "kraje.shp", "Data"], 1];
kraD = Import[NotebookDirectory[] <> "kraje.dbf", "LabeledData"];
kraD = Table[
  MapThread[Rule, {kraD[[All, 1]], kraD[[All, 2, n]]}, {n, Length[kraD[[1, 2]]}]];
okrM =
  Flatten["Geometry" /. Import[NotebookDirectory[] <> "okresy.shp", "Data"], 1];
okrD = Import[NotebookDirectory[] <> "okresy.dbf", "LabeledData"];
okrD = Table[
  MapThread[Rule, {okrD[[All, 1]], okrD[[All, 2, n]]}, {n, Length[okrD[[1, 2]]}]];
orpM =
  Flatten["Geometry" /. Import[NotebookDirectory[] <> "SO_ORP.shp", "Data"], 1];
orpD = Import[NotebookDirectory[] <> "SO_ORP.dbf", "LabeledData"];
orpD = Table[
  MapThread[Rule, {orpD[[All, 1]], orpD[[All, 2, n]]}, {n, Length[orpD[[1, 2]]}]];
obce = Import[NotebookDirectory[] <> "obce.shp", "Data"];
obeM = Flatten["Geometry" /. obce, 1];
obeD = Flatten["LabeledData" /. obce, 1];
obeD = Table[
  MapThread[Rule, {obeD[[All, 1]], obeD[[All, 2, n]]}, {n, Length[obeD[[1, 2]]}]];

```

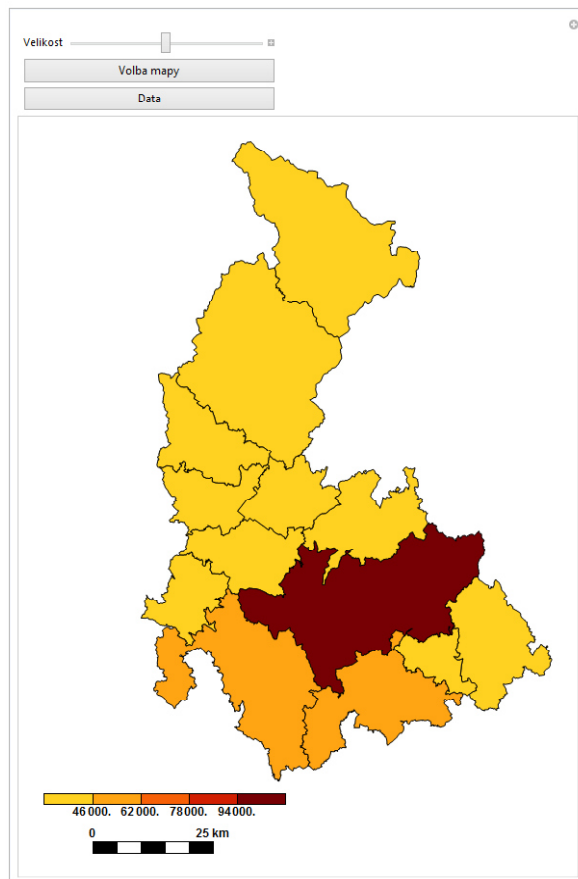


### Popis funkcí programu:

Po načtení potřebných souborů se po stisknutí “Volba mapy” otevře okno, ve kterém je třeba zvolit kraj, jehož mapu chceme zobrazit, popřípadě je možno zobrazit pouze mapu jednoho z krajů v daném okrese. Volbou konkrétní “Hranice” můžeme dále mapu rozdělit na obce s rozšířenou působností (volba “ORP”), na obce (volba “Obce”) či nechat pouze prázdnou mapu (volba “Blank”). Na levé straně okna se vykresluje zmenšenina naší zvolené mapy. Stiskem “OK” si necháme mapu vykreslit. Posuvníkem “Velikost” můžeme zmenšovat či zvětšovat zobrazovanou mapu. Stiskem tlačítka “Data” se zobrazí okno, kde můžeme zvolit data, která chceme na mapě zobrazit pro vybrané oblasti, a to buď jako kartogram nebo jako koláčové diagramy (záložky “Gradient” a “PieChart”).



Obr. 3 : Výběr mapy Olomouckého kraj, rozděleného na obce s rozšířenou působností



Obr. 4 : Počet obyvatel v obcích s rozšířenou působností v roce 1991 (kartogram)



## II Využití WolframAlpha v geografii

Jak jsme již uvedli v první kapitole, pro geovědy je důležitou funkcí programu možnost importovat data z neustále rozšiřované databáze matematických, vědeckých a socioekonomických informací, která je pro uživatele programu dostupná online prostřednictvím služby WolframAlpha. WolframAlpha funguje i samostatně jako „odpovídající stroj“. Výhodou pro uživatele je možnost zadávat otázky přirozeným jazykem (přesněji anglicky). Dotaz má velmi jednoduchou syntaxi WolframAlpha [“text dotazu”]. Nejdříve si ukážeme, jak si poradí s notoricky známou otázkou: “Jaká je odpověď na základní otázku života, vesmíru a vůbec?”:

In[198]= **WolframAlpha** [

**"Answer to the Ultimate Question of Life, the Universe, and Everything"]**

Out[198]=

Assuming The Ultimate Answer | Use [Monty Python's Meaning of Life](#) instead

Input interpretation: +

Answer to the Ultimate Question of Life, the Universe, and Everything

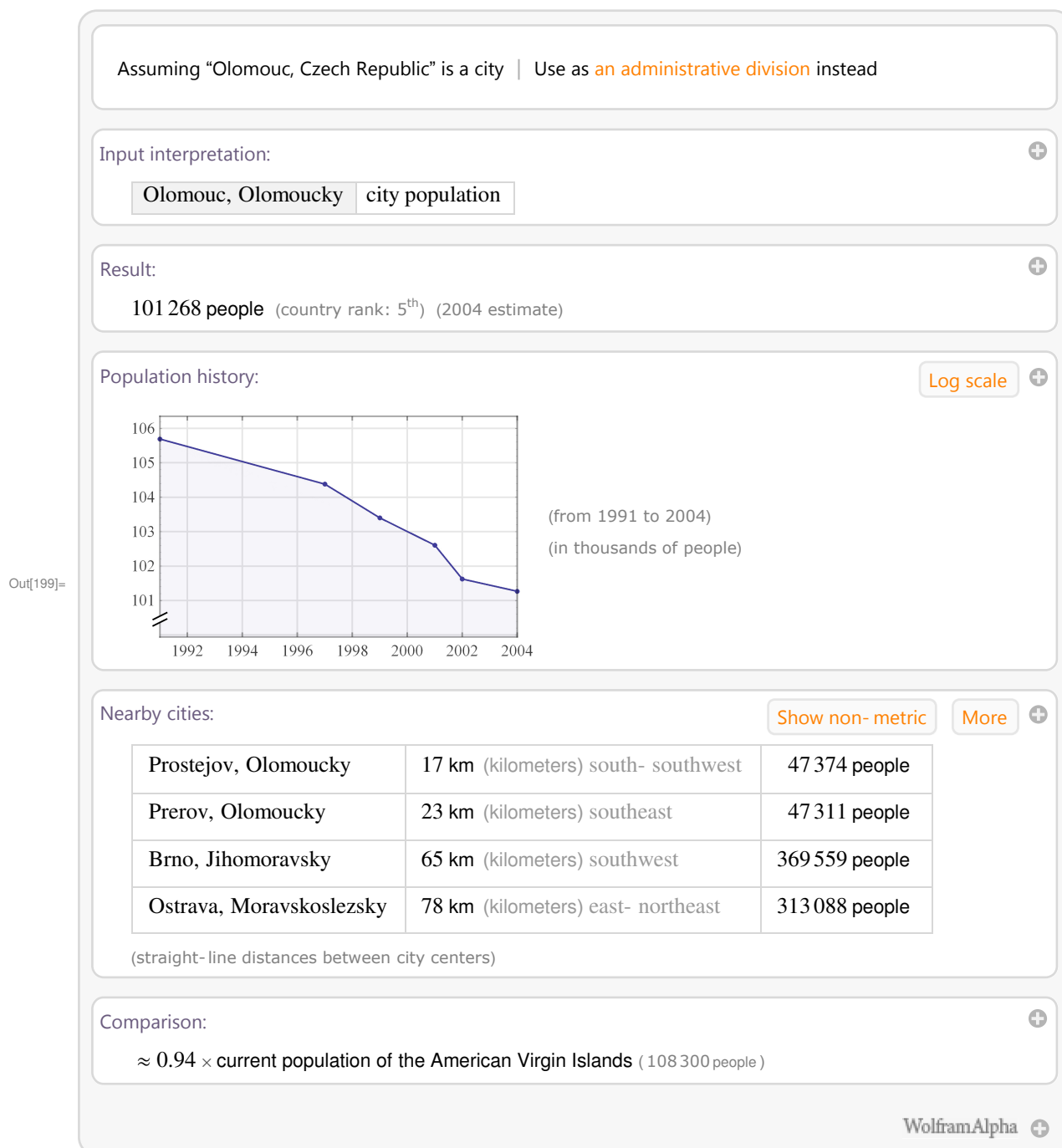
Result: +

42  
(according to Douglas Adams' humorous science-fiction novel *The Hitchhiker's Guide to the Galaxy*)

WolframAlpha +

Jak je vidět, s tímto “podrazem” si WolframAlpha poradil velmi dobře. Geografům se ale bezesporu bude více hodit např. dotaz na počet obyvatel Olomouce:

```
In[199]:= WolframAlpha["Population of Olomouc, Czech Republic"]
```



Jak je vidět z odpovědi, je primárně strukturována pro amerického uživatele, proto se snaží přiblížit počet obyvatel srovnáním s obyvatelstvem Amerických Panenských ostrovů. Využití odpovědi je také limitováno chybějícími údaji o zdrojích dat, nelze tedy posoudit jejich validitu ani kriticky komparovat různé zdroje. Nic ale nebrání použití WolframAlfa pro účely popularizační a prezentační. Ukážeme si několik možností:

WolframAlpha poskytuje i obrázky:

In[200]:= `WolframAlpha["flag of the Czech Republic"]`

Input interpretation: +

Czech Republic flag

Result: +

Out[200]=



WolframAlpha +

Bibliografické údaje:

In[201]:= `WolframAlpha["Václav Havel"]`

Assuming Vaclav Havel | Use [Václav Havel](#) instead


Input interpretation: +

Vaclav Havel (politician)

Basic information: +

full name	Vaclav Havel
date of birth	Monday, October 5, 1936 (78 years ago)
place of birth	Prague, Hlavní město Praha
date of death	Sunday, December 18, 2011 (age: 75 years) (3 years ago)
place of death	Vlcice, Kralovehradecky

Image: +



Leadership positions: +

official position	President	President (Czechoslovakia)
country	Czech Republic	Czech Republic
start date	2. 2. 1993	29. 12. 1989
end date	2. 2. 2003	20. 7. 1992

Out[201]=

Timeline: +Familial relationships: Show full dates +

## Parents:

Václav Maria Havel | Božena Vavrečková

## Sibling:

Ivan M. Havel

## Spouses:

Olga Havlová (1964–1996) | Dagmar Havlová (1997–2011)

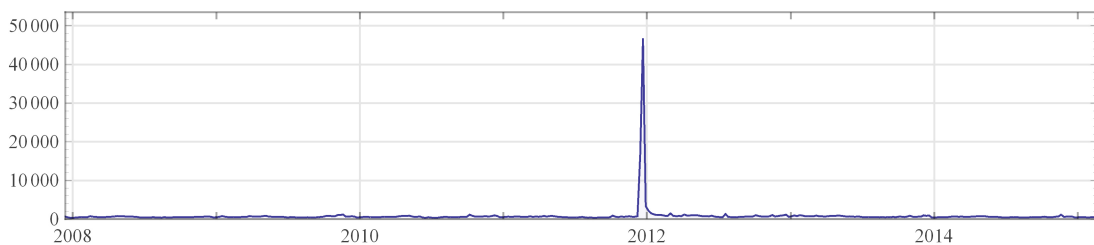
Notable films: +

## Appeared in:

Horem pádem (2004) | Czech–Made Man (2011)

[Definitions](#)Wikipedia summary: +

Václav Havel (5 October 1936 – 18 December 2011) was a Czech playwright, essayist, poet, philosopher, dissident and statesman. From 1989 to 1993, he served as the first democratically elected president of Czechoslovakia in 41 years. He then served as the first president of the Czech Republic (1993–2003) after the Czech–Slovak split. Within Czech literature he is known for his plays, essays, and memoirs.

[Full entry »](#)Wikipedia page hits history: Log scale +

(in hits per day)

(based on weekly averages of daily hits to English-language "Václav Havel" page)


Mapové podklady:

In[202]:= `WolframAlpha["map of Burundi"]`

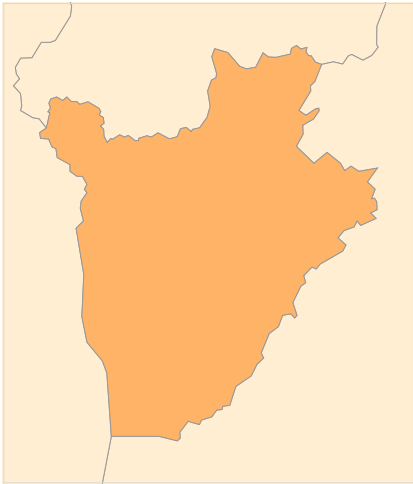
Input interpretation: +

Burundi location

Result: Show mesh +



Out[202]=



[Satellite image »](#)

WolframAlpha +

Data o počasí:

In[203]:= `WolframAlpha["Weather forecast for Olomouc, Czech Republic"]`

Input interpretation: +

weather forecast
Olomouc, Czech Republic

Weather forecast for Olomouc, Czech Republic: Show non-metric

More days
More details +

Tonight:

between 1 °C and 5 °C

clear (all night)

Tomorrow:

between 1 °C and 10 °C

clear (all day)

Tomorrow night:

between 3 °C and 6 °C

clear (all night)

Current forecast details: Next week | Show non-metric | More +

Temperature:

Day	High	Low
Mar 9		
Mar 10	10 °C	1 °C
Mar 11	6 °C	1 °C
Mar 12	5 °C	-1 °C
Mar 13	4 °C	-1 °C
Mar 14	6 °C	0 °C
Mar 15	7 °C	0 °C
Mar 16		

Cloud cover:

clear: 54.6% (3.7 days)
overcast: 0% (0 minutes)

Conditions:

snow: 29.6% (2 days)
rain: 23.1% (1.6 days)

Precipitation rate (mm/h):

Out[203]=

(with water equivalent of snow)

Wind speed (m/s):

Historical temperatures for March 9: [Show table](#) [Show non-metric](#) +

low: -10 °C 1996      average high: 7 °C      high: 19 °C 1975  
 average low: -1 °C

(daily ranges, not corrected for changes in local weather station environment)

Weather station information: [Show non-metric](#) [More](#) +

name	LKPO (Prerov Airport)
relative position	23 km SSE (from center of Olomouc)
relative elevation	(comparable to center of Olomouc)
local time	6:15:09 pm CET   Monday, March 9, 2015
local sunlight	sun is below the horizon azimuth: 269° (W)   altitude: -5°

[Units](#)  
[Satellite image »](#)

WolframAlpha +

Data o struktuře populace:

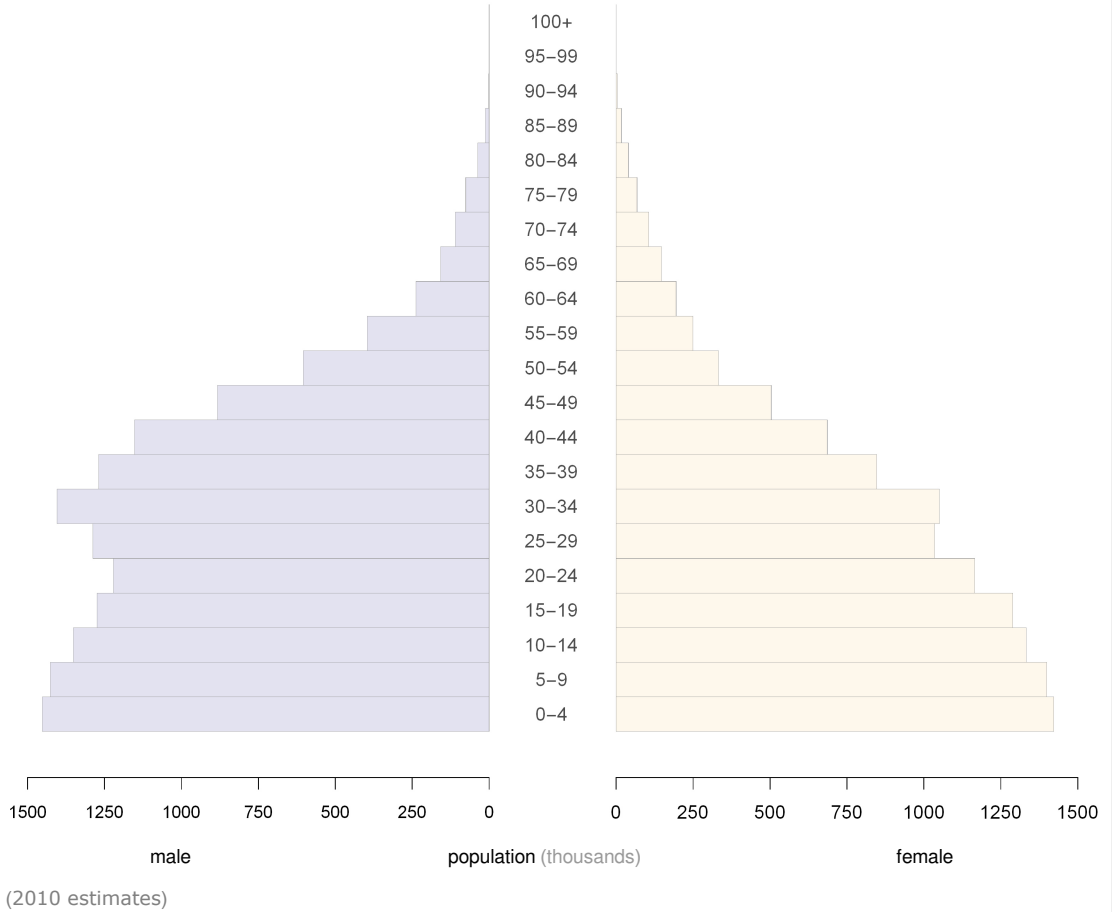
In[204]:= WolframAlpha["Saudi Arabia population distribution"]

Input interpretation:

Saudi Arabia age distribution

Result:

Out[204]=



Population by age and sex:

Show details

age	male	female	all
0 to 14	4.229 million people	4.154 million people	8.383 million people
15 to 64	9.729 million people	7.355 million people	17.08 million people
65+	395 000 people	383 000 people	778 000 people
all	14.35 million people	11.89 million people	26.25 million people

(2010 estimates)



Jaderné elektrárny v okruhu do 500 kilometrů od Vídně:

```
In[205]:= WolframAlpha["nuclear power reactors within 500 kilometers of Wien, Austria"]
```

Assuming "Wien, Austria" is a city | Use as an administrative division instead

---

Input interpretation: +

nuclear power reactors    within 500 km (kilometers) of Vienna, Austria

---

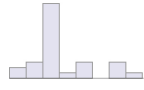
Nuclear power reactors within 500 km: More +

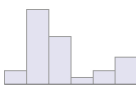
- Dukovany- 1 (97.7 km N)
- Dukovany- 2 (97.7 km N)
- Dukovany- 3 (97.7 km N)
- Dukovany- 4 (97.7 km N)
- Bohunice- 1 (101.7 km ENE)
- ⋮

(total: 27)

---

Basic information: +

nuclear reactor site	all	Bohunice A1 (nuclear power station)   Bohunice V1 (nuclear power station)   Bohunice V2 (nuclear power station)   Dukovany (nuclear power station)   Grafenrheinfeld (KKG) (nuclear power station)   Gundremmingen (nuclear power station)   Isar (nuclear power station)   Krško (nuclear power station)   Mochovce (nuclear power station)   Mochovce II (nuclear power station)   Niederaichbach (KKN) (nuclear power station)   Paks (nuclear power station)   Temelin (nuclear power station)
location	all	Bavaria, Germany   Jihocesky, Czech Republic   Nitriansky, Slovakia   Spodnjeposavska, Slovenia   Tolna, Hungary   Trnavsky, Slovakia   Vysočina, Czech Republic
net capacity	total	16 331 MW (megawatts) (electric power)
	median	471 MW (megawatts) (electric power)
	highest	1410 MW (megawatts) (electric power) (Isar- 2)
	lowest	100 MW (megawatts) (electric power) (Niederaichbach)
	distribution	
gross capacity	total	17 342 MW (megawatts) (electric power)
	median	500 MW (megawatts) (electric power)

highest	1485 MW (megawatts) (electric power) (Isar- 2)
lowest	106 MW (megawatts) (electric power) (Niederaichbach)
distribution	

Locations:

Show coordinates +



Out[205]=

Local map:

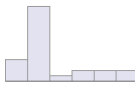
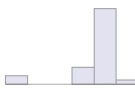

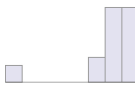
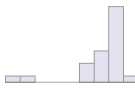
1700 kilometers across Non-metric +



(based on current OpenStreetMap data)

Power generation:

energy production	total	109 435 GW h/yr (gigawatt hours per year)
	median	3631 GW h/yr (gigawatt hours per year)
	highest	11 656 GW h/yr (gigawatt hours per year) (Isar- 2)
	lowest	0 GW h/yr (gigawatt hours per year) (Bohunice A- 1 and Gundremmingen A)

	distribution	
energy availability factor	median	88.8%
	highest	100% (Isar- 1)
	lowest	0% (Gundremmingen A)
	distribution	
load factor	median	88%
	highest	97.9% (Krško)
	lowest	2.7% (Niederaichbach)
	distribution	
time on line	total	166 906 h/yr (hours per year)
	median	7917 h/yr (hours per year)
	highest	8780 h/yr (hours per year) (Bohunice- 2)
	lowest	1128 h/yr (hours per year) (Niederaichbach)
	distribution	
operational factor	median	90.4%
	highest	100% (Bohunice- 2)
	lowest	22.2% (Niederaichbach)
	distribution	

## 12 Závěr

### Anotace

Text Software Mathematica pro geografii je koncipován jako učební materiál pro studenty geografických oborů, zejména Regionální geografie, Geografie (dvouoborové) a Mezinárodních rozvojových studií. Úvodní oddíly učebnice jsou věnovány představení programu a jeho základním funkcím, podstatná část textu pak představuje užití programu při řešení typických geografických problémů. Příklady použití jsou voleny s ohledem na potřeby studentů bakalářského a navazujícího studia.

### Závěr

Jednou z ambicí tvůrců programu Mathematica je zpřístupnění matematiky širokému okruhu „nematematické“ odborné veřejnosti. Tato koncepce činí z programu ideální nástroj pro geografii, kteří se ve svém výzkumu neobejdou bez statistických analýz a prostorových vizualizací, v jejich vzdělávání ale zpravidla na matematiku nezbývá mnoho prostoru. Text Software Mathematica pro geografii mapuje možnosti využití programu při řešení typických geografických úloh zhruba v rozsahu studia navazujících magisterských studijních oborů. Protože je cílen na začátečníky, věnuje prostor i základnímu představení programu, jeho funkcí a způsobů komunikace s jádrem programu. Autoři si rozhodně nekladli za cíl vytvořit vyčerpávající přehled všech možností použití programu, který se ostatně sám neustále vyvíjí a v nových verzích stále přibývají nástroje propojující program Mathematica se softwarem geografických informačních systémů (GIS).

### Summary

One of Mathematica program creators' ambitions is to make mathematics accessible to a wide range of a "non-mathematical" professional community. This concept makes the program an ideal tool for geographers who cannot do their research without statistical analysis and spatial visualization. But there is no much room left for mathematics in their educational curriculum. Software Mathematica text for geographers maps possible program usage for solving typical geographical tasks around the scope of Master degree study. Being targeted at the beginners, it presents the basic performance of the program, its function and ways of communicating with the core of the program. The authors of the program were certainly not focused on creating the exhausted list of all its possible uses. The program is being constantly developed and the new versions of it are characterized by constantly growing number of tools that link Mathematica program with the Geographical Information Systems software (GIS).

## 13 Použitá literatura

- Baumann, G.: Mathematica for Theoretical Physics. Springer Science, New York. 2005.
- Dick, S., Riddle, A., Stein, D.: Mathematica in the Laboratory. Cambridge University Press, Cambridge. 1997.
- Hassani, S.: Mathematical methods using Mathematica: for students of physics and related field. Springer Science, New York. 2003.
- Chramcov, B.: Základy práce v prostředí Mathematica. Univerzita Tomáše Bati ve Zlíně, Zlín. 2005.
- Jirásek, F., Čipera, S., Vacek, M.: Sbíрка řešených příkladů z matematiky II. SNTL, Praha. 1989.
- Jirásek, F., Kriegelstein, E., Tichý, Z.: Sbíрка řešených příkladů z matematiky. SNTL/ALFA, Praha. 1982.
- Kopáček, J. a kol.: Příklady z matematiky pro fyziky III. MATFYZPRESS MFF UK, Praha. 1996.
- Kováčová, M., Záhonová, V.: Matematika pomocou THE MATHEMATICAL EXPLORER. Slovenská technická univerzita v Bratislave, 2006.
- McMahon, D., Topa, D.: A Beginner's Guide to Mathematica. Chapman & Hall/CRC, New York. 2006.
- Wellin, P., Gaylord, R., Kamin, S.: An Introduction to Programming with Mathematica. Cambridge University Press, Cambridge. 2008.

Homepage of Wolfram Research <http://www.wolfram.com>.

RNDr. Miloš Fňukal, Ph.D. a kolektiv

Software Mathematica pro geografy

Výkonný redaktor Prof. RNDr. Zdeněk Dvořák, DrSc., Ph.D.

Odpovědná redaktorka Mgr. Jana Kreiselová

Technická redakce autor

Určeno pro studenty a akademické pracovníky VŠ

Vydala a vytiskla Univerzita Palackého v Olomouci

Křížkovského 8, 771 47 Olomouc

[www.upol.cz/vup](http://www.upol.cz/vup)

e-mail: [vup\(at\)upol.cz](mailto:vup(at)upol.cz)

Olomouc 2015

1. vydání

z. č. 2015/0079

Edice - Odborné publikace

ISBN 978-80-244-4472-7

Neprodejná publikace