Otazníky středoškolské informatiky

DALIBOR MARTIŠEK

Vysoké učení technické, Brno

Říká se, že ve využívání moderních technologií lze vystopovat dva základní směry: směr čistě uživatelský, „amatérský“ a směr „profesionální“. Ve stejné situaci je prý i výpočetní technika. Středoškolská informatika dnes řeší problém, jak v této situaci vychovávat své maturanty, a musí si přitom klást řadu otázek.

1 Má informatika na výběr dva směry?

Uživatelský směr zacházení s moderní technologií lze demonstrovat třeba na příkladu mobilního telefonu. Mohu volat, posílat SMSky, fotografovat, nahrávat jednoduchá videa. Nepotřebuji vědět, jak to funguje. Je to pro mě černá skříňka. Telefon prostě používám. Pak se porouchá a nastoupí profesionál. Zjistí závadu, vymění klávesnici, (displej, přeinstaluje software…). Počítač je dnes totéž. Spotřební elektronika, kterou mohu používat bez znalostí principů jeho činnosti. Psát dopisy, posílat poštu, vést účetnictví. Pak něco přestane fungovat a musí nastoupit odborník. Diagnostikuje a vymění vadnou komponentu, nakonfiguruje vzdálenou tiskárnu…

Podle mého názoru se středoškolská informatika topí v problému, zda vychovávat „amatéry“, tj. uživatele výpočetní techniky, anebo „profesionály“, tj. její údržbáře a opraváře. Svědčí o tom některé aktuální maturitní otázky a reakce na ně. Uvedu příklad. Maturitní otázka zní:

Číslo vyjádřené ve dvojkové soustavě jako 0011 1011 0100 odpovídá číslu:

A) 1674 v oktalové soustavě C) 1654 v oktalové soustavě

B) 958 v desítkové soustavě D) 3B4 v hexadecimální soustavě

Zajímavější než otázka sama jsou reakce odpůrců a zastánců takto postavené maturity:

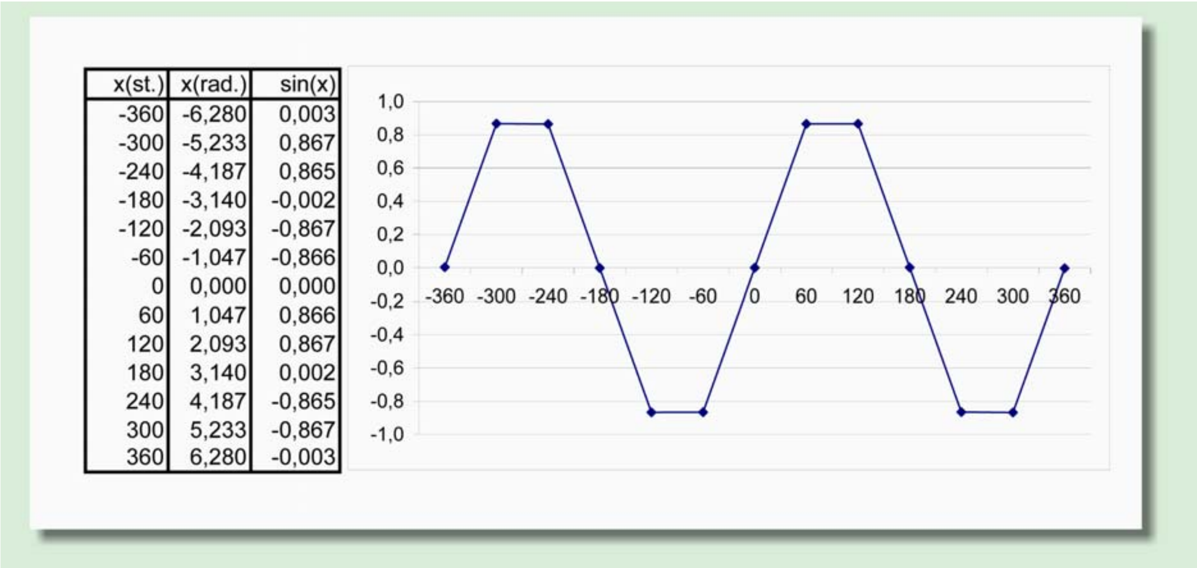
Odpůrci říkají: „Pro běžného uživatele (např. pro sekretářku, učitele při práci na své přípravě na hodinu, internetového surfaře,...) postrádá tato úloha jakýkoli smysl. Při běžném používání počítače jsou tyto znalosti potřeba asi tolik, jako lékařský titul při poskytování první pomoci“. Propagují maturitu z informatiky, která spočívá ve vytvoření typograficky správné pozvánky na oslavu narozenin a její rozeslání e-mailem.

Zastánce tvrdí: „Ten, kdo **maturuje** z informatiky, není **řadovým** uživatelem. Lze předpokládat, že pokud nebude pokračovat na vysoké škole, stane na nějakém nižším správcovském postu, či jako programátor“. Propaguje oktalovou soustavu, DNS servery a sítě LAN.

Takže vychovávat „běžného uživatele“, anebo správce –„opraváře“? Obávám se, že ani jeden směr není správný. První z nich skutečně degraduje maturanta z ICT na někoho, kdo používá počítač jako každou jinou spotřební elektroniku. Mám-li použít výše uvedené zdravotnické přirovnání, úkolem maturity na zdravotní škole je vychovat zdravotní sestru, která nemá dělat nic jiného, než poskytovat laickou první pomoc. Druhé tvrzení je ovšem taky špatně. Chce totiž naopak po této zdravotní sestře, aby byla uměla vyoperovat žlučník.

2 Kam se poděla krása poznání?

Dovolte, abych tuto kapitolu začal příkladem ze své praxe. Každoročně cvičím pět šest studijních skupin v prvním ročníku z počítačové grafiky. Loni to bylo asi sto lidí, z nichž asi čtyřicet mělo maturitu z informatiky. Na začátku výuky jsem zadal úkol: za hodinu chci na monitoru počítače vidět graf funkce sinus. K dispozici čtvero programovacích prostředí, dva systémy matematického software, kompletní instalace MS Office a spojení s celým světem. Jeden student sestrojil „sinusoidu“ v MS Excel (viz obr. 1). Přesně tak ji prý v informatice dělali. Na otázku, kolik je sinus tři sta šedesáti stupňů, odpověděl správně. Na otázku, proč má tedy v tabulce hodnotu $-0,003$, odpovědět nedokázal. Reakce na otázku, proč má křivku tak hranatou, mě vyděsila. Menší krok by prodloužil tabulku. A ta nemůže být delší – nevešla by se totiž na stránku a byla by typograficky špatně… Přežil jsem svůj první infarkt a „řešení“ jsem uznal.

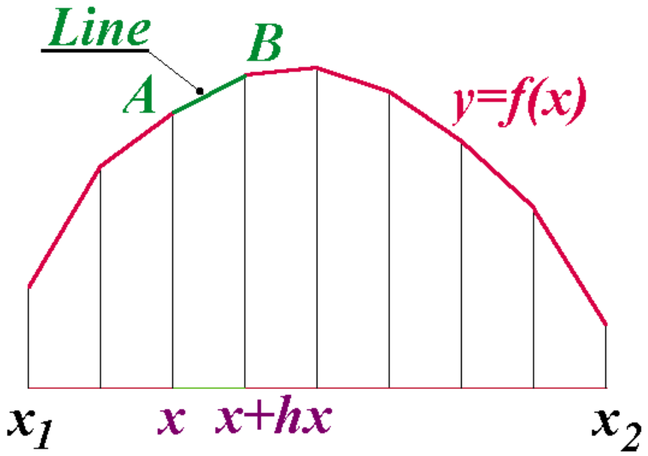


x(st.)	x(rad.)	sin(x)
-360	-6,280	0,003
-300	-5,233	0,867
-240	-4,187	0,865
-180	-3,140	-0,002
-120	-2,093	-0,867
-60	-1,047	-0,866
0	0,000	0,000
60	1,047	0,866
120	2,093	0,867
180	3,140	0,002
240	4,187	-0,865
300	5,233	-0,867
360	6,280	-0,003

Obr. 1.: Graf funkce $f(x) = \sin x$ v představách dnešního maturanta z informatiky

Dva studenti nakreslili na papír něco, co vzdáleně připomínalo soustavu jakýchsi půlkružnic, a sháněli se po skeneru. I toto „řešení“ jsem uznal. Další dva našli křivku někde na internetu a jeden ji (světe, div se!) naprogramoval. Můj úkol tak zvládlo šest – slovy ŠEST – lidí ze sta. Přitom devětadevadesáti lidem, kteří o programování neměli dosud ani ponětí, jsem tento konkrétní příklad vysvětlil docela snadno a společně jsme ho naprogramovali během zbývajících dvaceti minut. Algoritmus je zřejmý z obr. 2, procedura v programu Borland Delphi vypadá takto:

```
Procedure Graf_Funkce(Sender:TObject);
      Function f(x:Real):Real
      Begin
         f:=sin(x)
      End;
Begin
 x:=x1;
 Repeat
      A[1]:=x;A[2]:=f(x);x:=x+hx;
      B[1]:=x;B[2]:=f(x);
      Line(A,B,clGreen);
 Until x>x2;
End;
```



Obr. 2.: Princip algoritmu konstrukce grafu spojité funkce

A když se na závěr po přidání ještě jednoho cyklu sinus rozvlnil a začal „pochodovat“, bylo z toho sto dospělých lidí na začátku jednadvacátého století doslova u vytržení. Nad obyčejnou „živou křivkou“ jásali daleko víc než nad nějakými efektními triky počítačových her.

Asi poprvé v životě totiž vytvořili něco opravdu sami. Snad poprvé v životě zažili krásu poznání. Tedy něco, co je v obou výše uvedených přístupech (a obávám se, že zřejmě v celé středoškolské informatice) většinou naprosto ignorováno.

3 Tlačit klavír k židli, nebo židli ke klavíru?

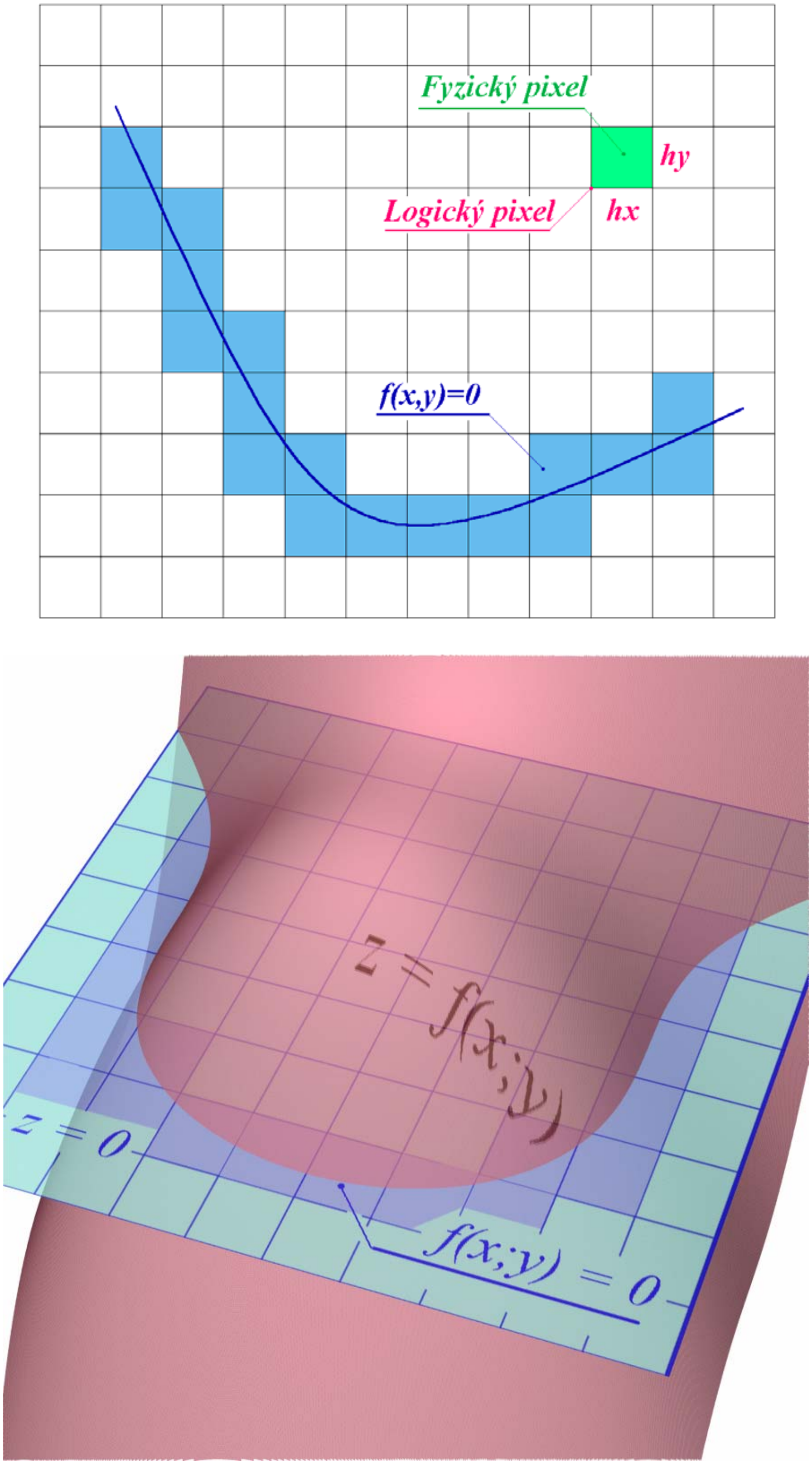
Další otazník dnešní středoškolské informatiky. ICT jsou většinou v duchu „uživatelského“ přístupu ztotožňovány s obsluhou počítače. Ba co hůře: s obsluhou softwaru firmy Microsoft. Znám střední školy, kde je produktům této firmy věnováno dva a půl roku, a to údajně proto, že jsou nejrozšířenější. Něco na tom jistě je. Excel jistě může dobře posloužit v řadě kancelářských prací, Word při editaci dokumentů. Škola by se však měla omezit na nejnutnější seznámení s tím, že vše ostatní je čloněk schopen v případě potřeby zvládnout sám – podobně jako třeba obsluhu mobilního telefonu.

Dnešní glorifikace kancelářského software může být prostým důsledkem toho, že na jiný software nemá škola peníze nebo učitele. Jde ovšem tak daleko, že se v něm provozují věci, které z něj lze sice jistým způsobem vymačkat, ale ke kterým se příliš nehodí. Na jedné straně je přirozené, že se někteří uživatelé Excelu pokoušejí i o úlohy, ke kterým není primárně určen (např. snahu o co největší „přiblížení“ grafu funkce $y = \operatorname{tg} x$ k asymptotě), na straně druhé bych ale očekával,že výsledkem těchto pokusů bude konstatování, že na ilustraci podobných jevů Excel zrovna moc vhodný není.

Navíc lze v tomto případě nalézt daleko snažší, sofistikovanější a elegantnější řešení. To využívá známé skutečnosti, že „body“ na výstupním zařízení počítače nejsou „bezrozměrné“, ale každý z nich představuje jakousi „elementární plošku“ (pixel), který se modeluje většinou jako čtverec:

Obr. 2.: Pixely výstupního zařízení počítače

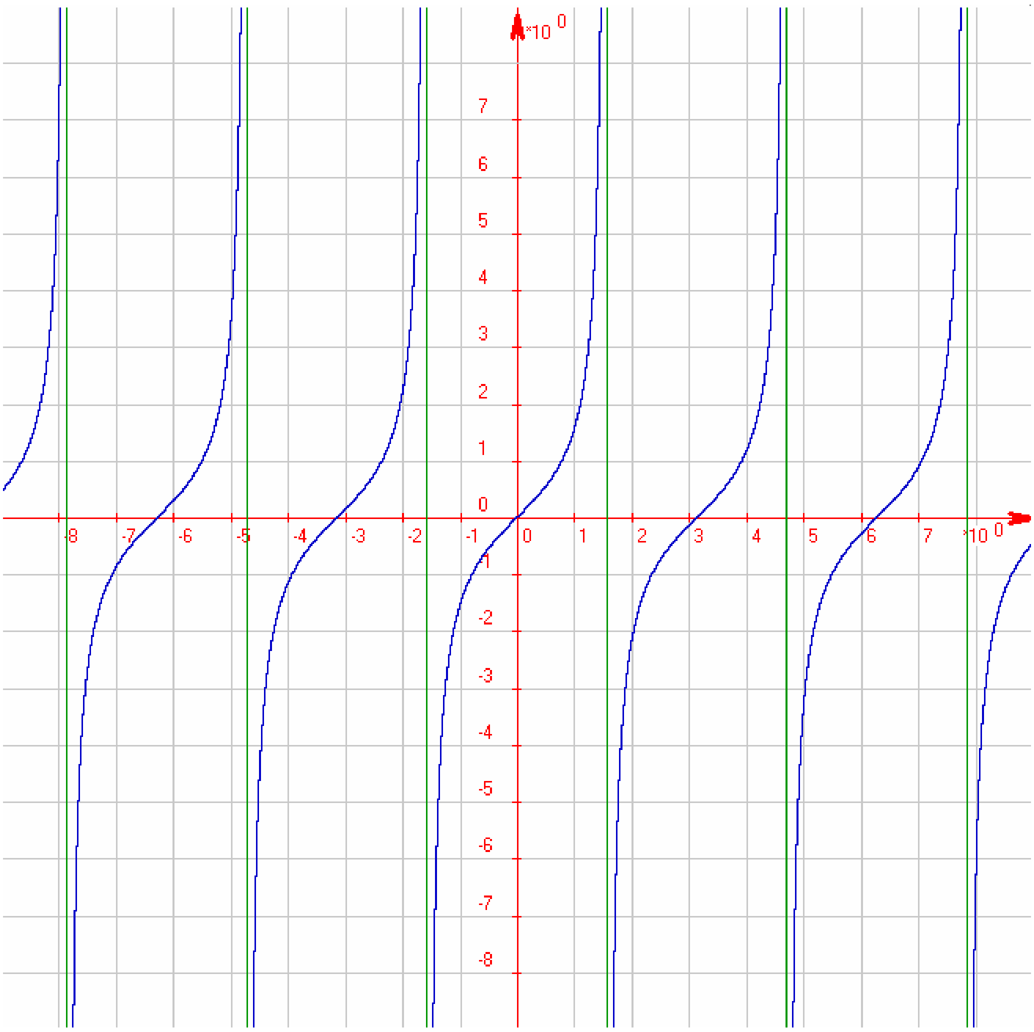
Křivka vykreslená na takovém zařízení se skládá z nepřekrývajících se čtverců tak, jak je znázorněno na dalším obrázku. Základní myšlenka konstrukce křivky tedy spočívá v tom, že projdeme výstupní zařízení pixel po pixelu u každého z nich rozhodeme, zda bude obarven či nikoli. Kriteriem pro obarvení pixelu jsou přitom různá znaménka funkce $z = f(x; y)$



Obr. 3.: Princip konstrukce křivky typu $f(x;y)=0$

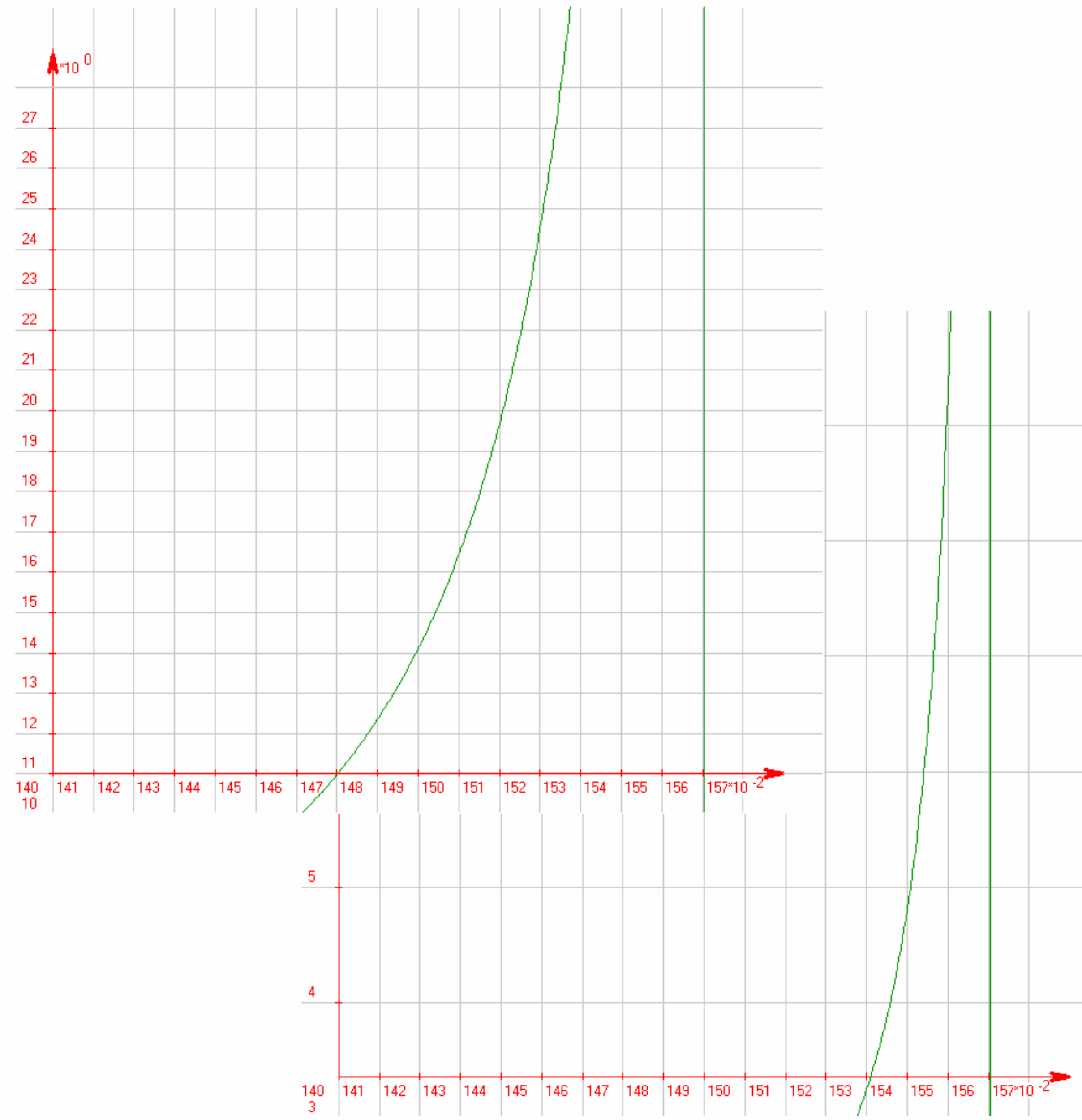
Nejjednodušší zdrojový kód programu v Borland Delphi vypadá takto:

```
procedure TForm1.Implicit_Krivka(Sender: TObject);
const x1=-5;x2=5;y1=-5;y2=5;
var   x,y,hx,hy:Real;
      i,j       :Integer;
Function f(x,y:real):real;
begin
f:=y-sin(x)/cos(x);
end;
begin
     hx:=(x2-x1)/Image1.Width;hy:=(y2-y1)/Image1.Height;
     x:=x1;i:=0;
     Repeat
        y:=y2;j:=0;
        Repeat
           if (f(x,y)*f(x+hx,y)<0) or (f(x+hx,y)*f(x+hx,y+hy)<0)
              then Image1.Canvas.Pixels[i,j]:=0;
           inc(j);y:=y-hy;
        until y<y1;
        inc(i);x:=x+hx;
     until x>x2;
end;
```



Obr. 4.: Funkce $y = \mathrm{tg}\, x$ **(výstupy procedury `Implicit_Krivka` v různých výřezech doplněné souřadnými osami)**

Přesnost grafu je omezena jen přesností výstupního zařízení (v tomto případě monitoru). Tento výstup je na obr. 4 doplněn jednoduchými souřadnými osami, které je jistě schopen doplnit každý zručnější programátor. Pouhou změnou výřezu okna můžeme pořídit libovolný detail přibližování křivky ke svislé asymptotě – viz obr. 5.



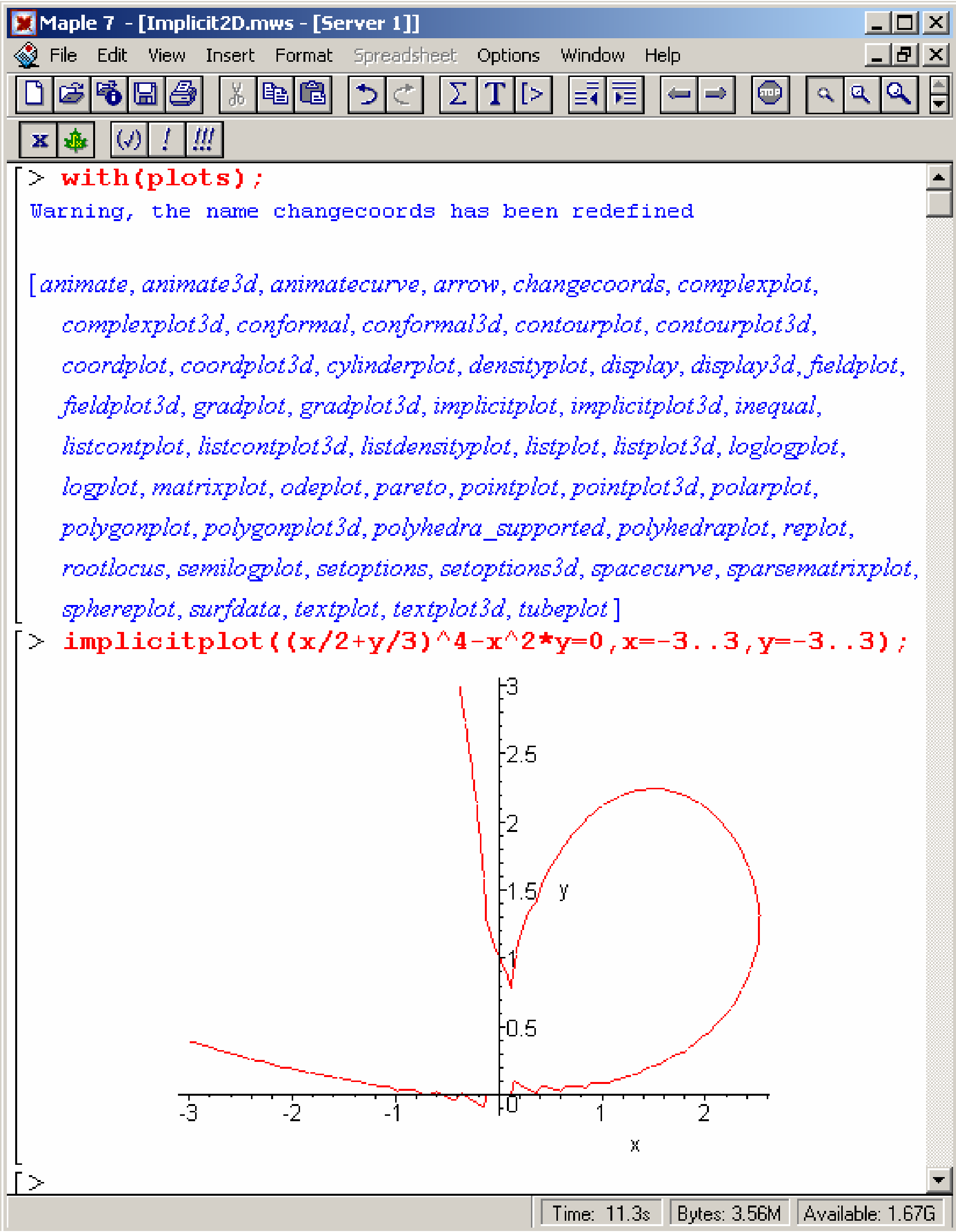
Obr. 5.: Funkce $y = \mathrm{tg}\, x$ - různé výřezy výstupního okna demonstrují přibližování grafu k asymptotě.

A nejen to. Jestliže do deklarace funkce zapíšeme např.

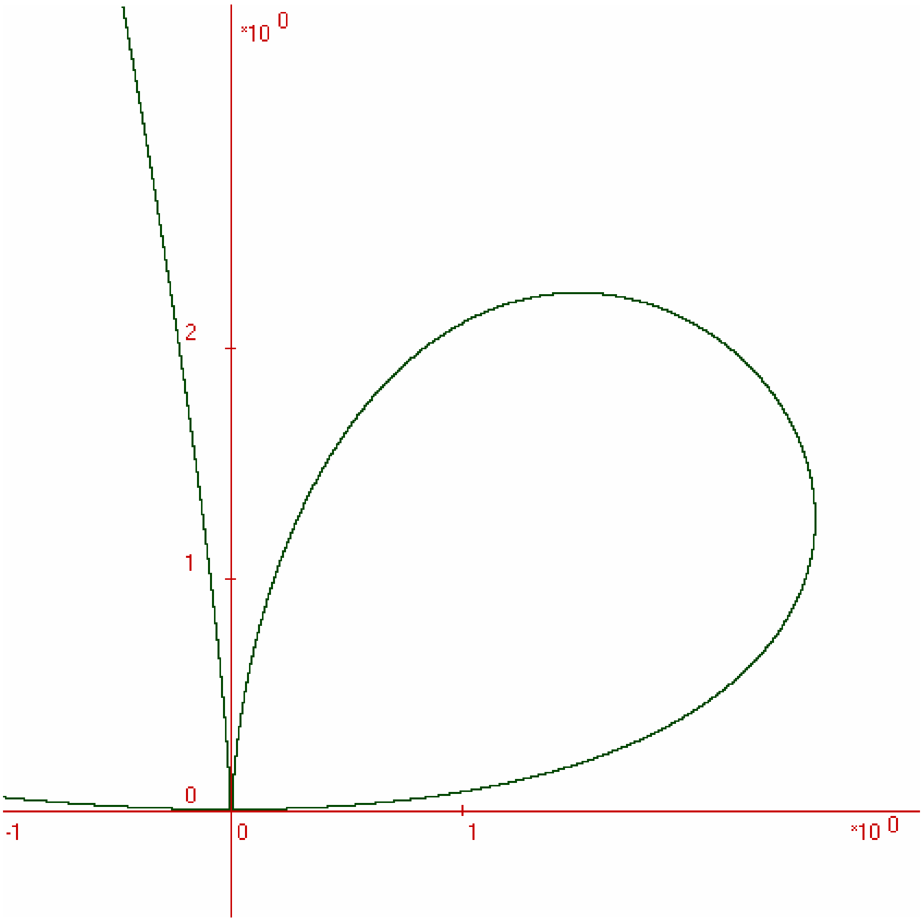
```
f:=sqr(sqr(x/2+y/3))-x*x*y,
```

procedura vykreslí křivku, kterou už Excelem nezvládneme docela určitě. V této kvalitě ji totiž neudělá ani specializovaný a velmi drahý matematický software.

Na střední škole může být procedurka Implicit_Krivka dobrou učební pomůckou, kterou vytvoří kantor. Na škole vysoké je možné prozradit jen algoritmus a chtít po studentech, aby ji napsali sami. Stále se totiž najdou takoví, kteří jsou toho schopni (i když jich bohužel rychle ubývá). Při srovnání obrázků 6 a 7 si pak můžeme spolu se studenty hrdě říci: „Dámy a pánové, instalace balíku Maple 11 do této učebny stála fakultu tři sta padesát tisíc. Jak sami vidíte, my jsme lepší!“ Značná nadsázka, samozřejmě. Ale v danou chvíli silně motivující pravda. Další velmi důležitý a bohužel značně opomíjený aspekt vzdělávacího procesu!



Obr. 6.: Křivka $\left(\frac{x}{2}+\frac{y}{3}\right)^4 - x^2 y = 0$ v podání firmy MapleSoft.



Obr. 6.: Křivka $\left(\frac{x}{2}+\frac{y}{3}\right)^4 - x^2 y = 0$ sestrojená procedurou `Implicit_Krivka`.

Studenti, kteří v informatice vidí jen klikání na ikony a tahání vzorečků po tabulkách, takový pocit hrdosti nad svým výsledkem nikdy nezažijí. Z programování mají hrůzu, a tak jejich semestrální práce s nadpisem „Výpočet integrálu složenou lichoběžníkovou metodou" obsahují namísto deseti řádků nějakého rozumného kódu deset stránek excelovských tabulek. A já se je dnes a denně snažím posílat od klavíru k židli. Většinou už bohužel zcela marně.

Konstrukce jednoduchých parametricky zadaných křivek a křivek zadaných v polárních souřadnicích je zcela analogická konstrukci grafů spojité funkce:

Graf funkce:	**Parametr. zadaná křivka**	**Křivka v polárních souřadnicích**
`...`	`...`	`...`
`A[1]:=x;A[2]:=f(x);`	`A[1]:=f(t);A[2]:=g(t);`	`A[1]:=rho(t)*cos(t);`
		`A[2]:=rho(t)*sin(t);`
`x:=x+hx;`	`t:=t+ht;`	`t:=t+ht;`
`B[1]:=x;B[2]:=f(x);`	`B[1]:=f(t);B[2]:=g(t);`	`B[1]:=rho(t)*cos(t);`
`Line(A,B,clGreen);`	`Line(A,B,clGreen);`	`B[2]:=rho(t)*sin(t);`
…	…	`Line(A,B,clGreen);`

Viděl jsem článek určený středoškolským učitelům, který se zabýval algoritmem vyčíslování hodnoty zadaného aritmetického výrazu, kde byl tento algoritmický problém povzbudivě označen jako „zajímavé cvičení pro mírně pokročilé programátory". Kéž by aspoň polovina maturantů byla takto „mírně pokročilá"! Pokud by tomu tak bylo, pak by ani zobecnění tohoto řešiče nemuselo být velkou překážkou. Takto zobecněný algoritmus může pak zpracovávat téměř libovolný matematický výraz. Tímto parserem se u nás zabýváme se studenty některých specializovaných oborů a jistě by ho zvládl i středoškolský učitel informatiky. A propojí-li ho příjemným uživatelským rozhraním s výše uvedenými procedurami na sestrojování křivek, dospěje k jednoduchému programu, který mohou studenti využít v matematice, fyzice a dalších předmětech. Celé systémy křivek mohou pak být „vyráběny“ jediným vzorečkem, který netřeba „tahat po tabulkách“.

Sestavením několika semestrálních prací studentů 1. ročníku oboru Matematické inženýrství na FSI VUT v Brně vznikl program Rovinne_Krivky. Vykresluje grafy funkcí, křivky zadané parametricky, polárně a implicitně. Celý program si zájemce může vyzkoušet zde.

4 Může matematika pomoci informatice?

Při výuce informatiky studenti často vytvářejí nejrůznější dokumenty. Jejich obsah závisí jen na fantazii vyučujícího a často zůstává jen u nejrůznějších pozvánek na ples či oslavu narozenin. Přitom předměty jako matematika či fyzika mohou poskytnout řadu námětů, při kterých musí student prokázat nejen potřebné znalosti textového editoru, ale i zpracovávaného tématu. Takové zadání může na nižším stupni gymnázia vypadat např. takto:

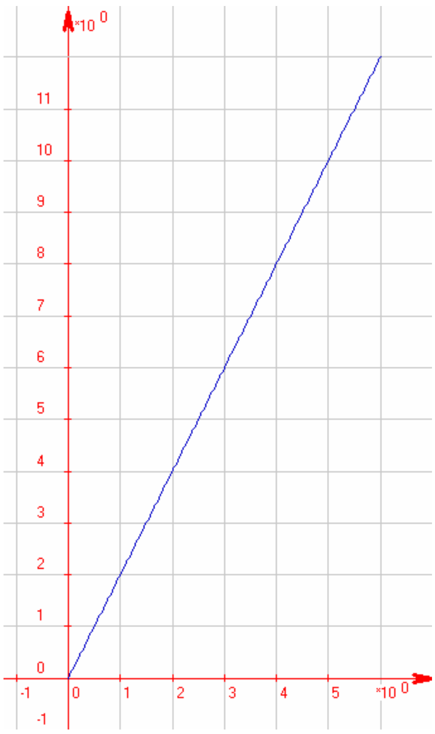
Přímá a nepřímá úměrnost – nákup rohlíků; doba jízdy

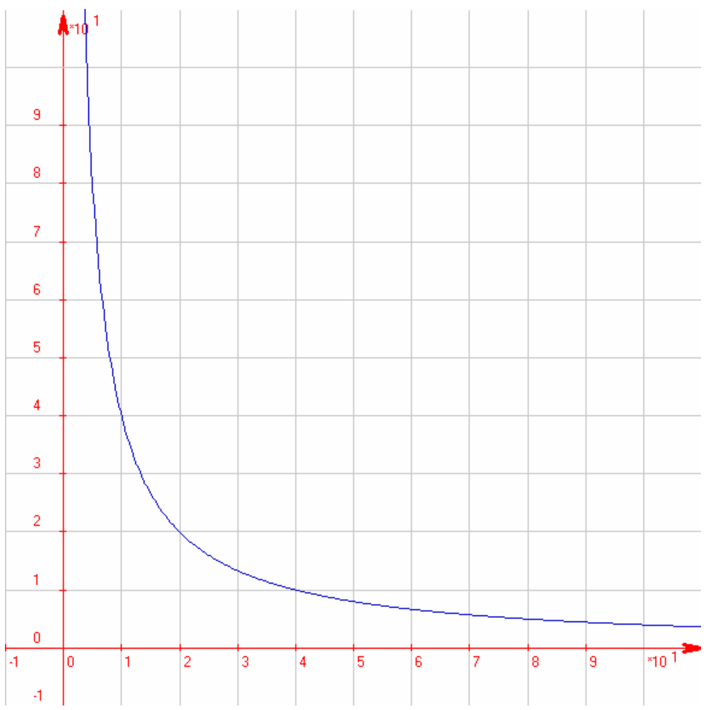
Počet rohlíků (x)	0	2	4	6	8	10
Cena (y)	0		8			

Rychlost (kmh^{-1})	5	10	20	40	80
Čas (hod.)			20		

Doplň tabulky, zapiš rovnici závislosti ceny rohlíků na jejich počtu a doby jízdy na rychlosti. Grafy vytvoř v programu Rovinne_Křivky, ulož a vlož do dokumentu. Zapiš obecnou rovnici přímé a nepřímé úměrnosti.

Řešení:





Počet rohlíků (x)	0	2	4	6	8	10
Cena (y)	0	4	8	12	16	20

Rychlost (kmh^{-1})	5	10	20	40	80
Čas (hod.)	80	40	20	10	5

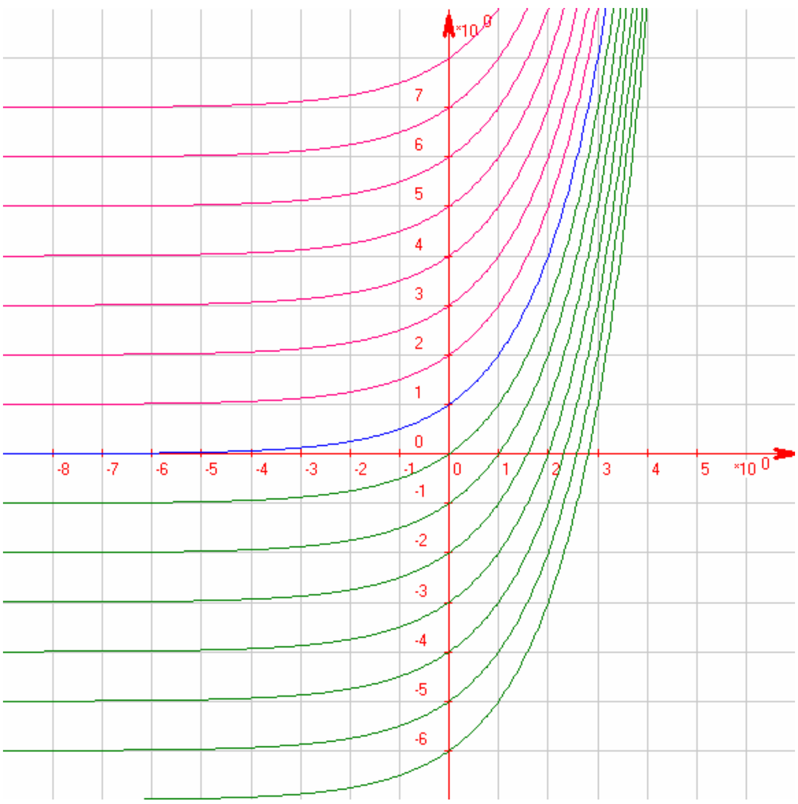
$y = 2 \cdot x$; obecně $y = k \cdot x$ $y = \frac{400}{x}$; obecně $y = \frac{k}{x}$

Analogicky pro vyšší stupeň gymnázia:

Funkce (rovnice, definiční obory, obory hodnot, grafy)

1. V programu Křivky vytvoř následující grafy a přenes do dokumentu.
2. Popiš podle vzoru.
3. Ve zdrojovém kódu programu Krivky vyhledej část kódu, která grafy sestrojuje a přenes do dokumentu.
4. Pokus se kód pochopit a vytvoř čárkovaný graf.

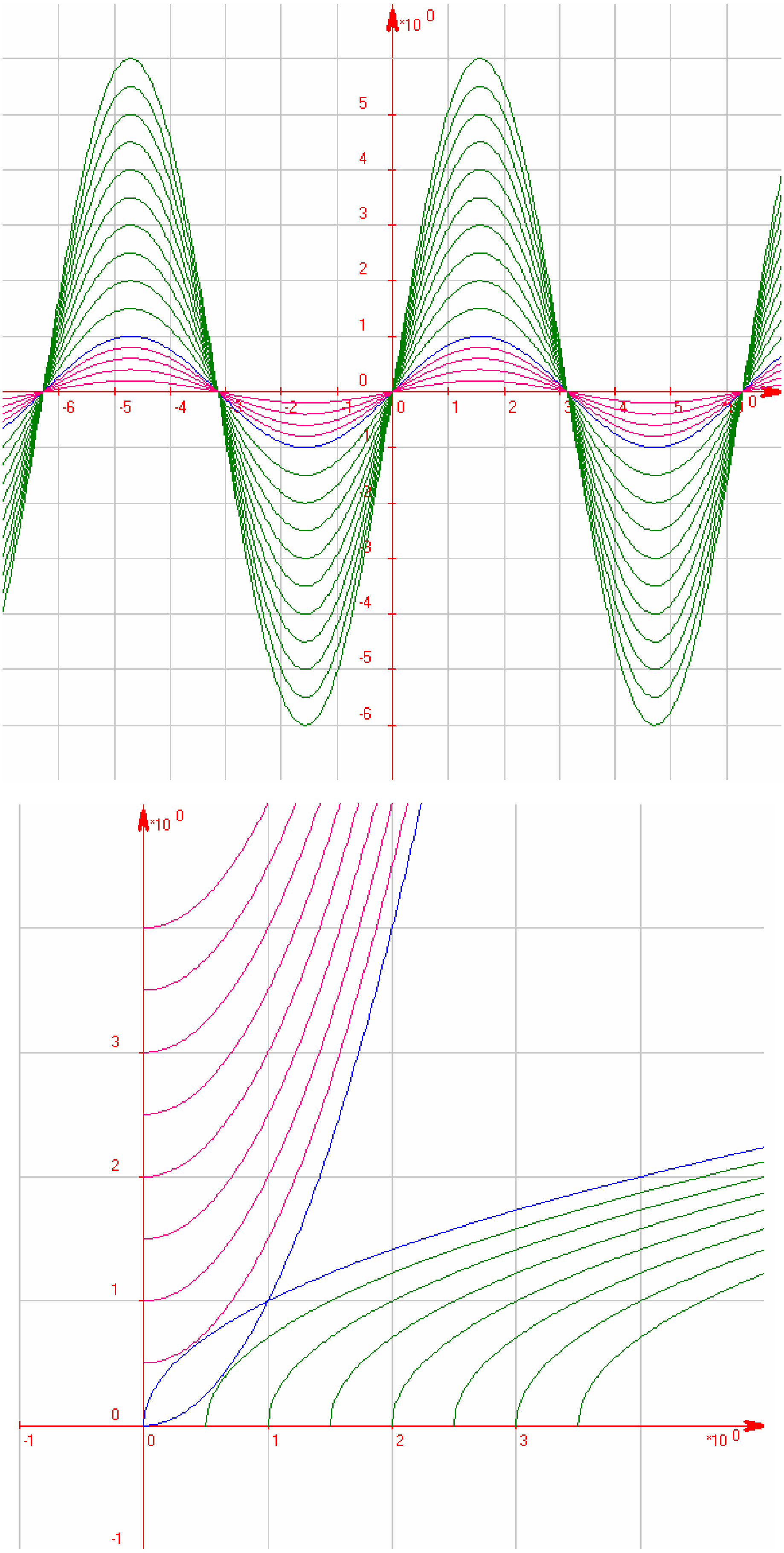
Vzor:



$f(x) = 2^x$
$D(f) = \mathbb{R}$; $H(f) = \langle 0; \infty)$

$f(x) = 2^x + c$; $c > 0$
$D(f) = \mathbb{R}$; $H(f) = \langle c; \infty)$

$f(x) = 2^x + c$; $c < 0$
$D(f) = \mathbb{R}$; $H(f) = \langle c; \infty)$



Obr. 7. Grafy pro samostatnou práci studentů dle předchozího vzoru

Vyhledání a doplnění kódu:

```
x:=x1;
Repeat
     A[1]:=x;A[2]:=Calc(EditFce.Text,ErrorReport);
     x:=x+hx;
     B[1]:=x;B[2]:=Calc(EditFce.Text,ErrorReport);
     Line(A,B,Red,Green,Blue);
Until x>x2;
```

Čárkovaný (přerušovaný) graf:

```
x:=x1;
Repeat
     A[1]:=x;A[2]:=Calc(EditFce.Text,ErrorReport);
     x:=x+hx;
     B[1]:=x;B[2]:=Calc(EditFce.Text,ErrorReport);
     x:=x+hx;
     Line(A,B,Red,Green,Blue);
Until x>x2;
```

5 Může informatika pomoci fyzice?

Uveďme opět několik příkladů.

Brownův pohyb: V biologii lze Brownův pohyb pozorovat v mikroskopu. Ve fyzice lze vysvětlit:

Hmotnost pozorované částice	M
Hmotnost molekul vody	m
Rychlost i-té molekuly vody	$\mathbf{v}_i$ - náhodná velikost i směr.
Hybnost i-té molekuly vody	$\mathbf{p}_i = m\mathbf{v}_i$

Hybnost molekul, které v daném časovém okamžiku narazí do pozorované částice

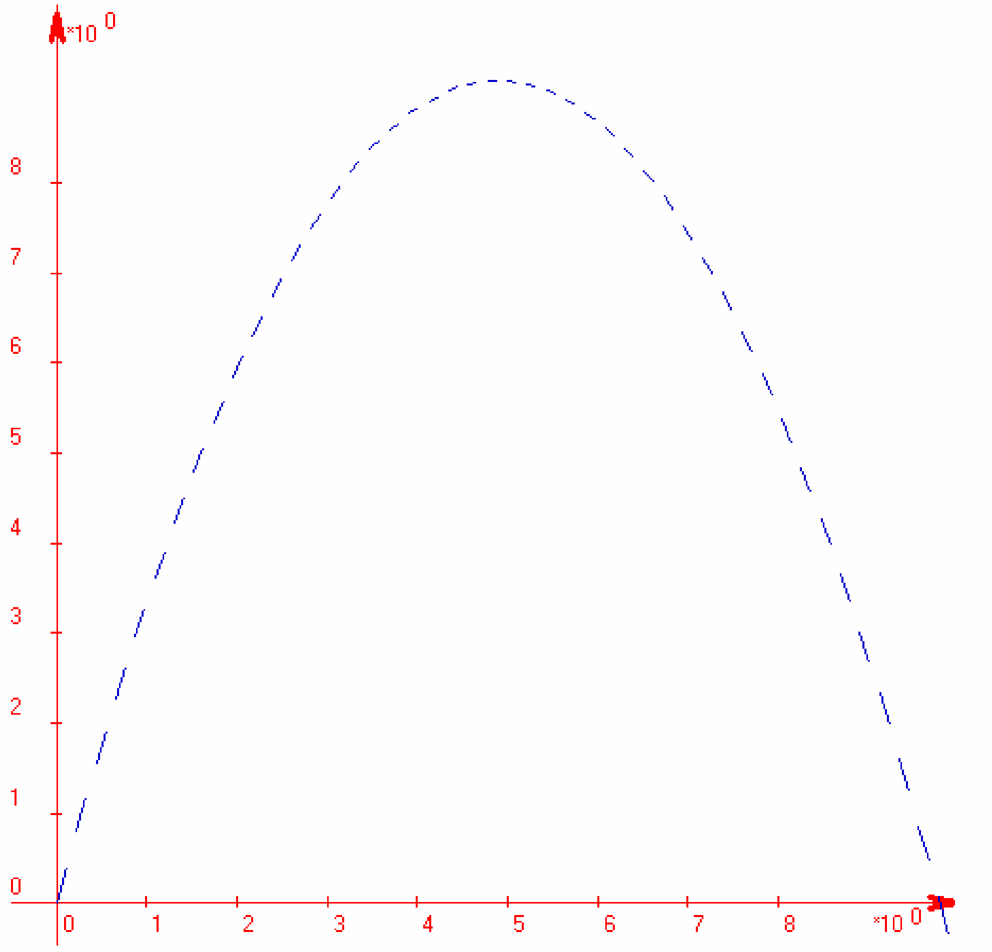
$$\mathbf{P} = m\mathbf{v}_1 + m\mathbf{v}_2 + \ldots + m\mathbf{v}_n .$$

Tuto hybnost předají pozorované částici, tj.: $M\mathbf{v} = m\left(\mathbf{v}_1 + \mathbf{v}_2 + \ldots + \mathbf{v}_n\right)$

Rychlost pozorované částice v daném okamžiku: $\mathbf{v} = \frac{m}{M}\left(\mathbf{v}_1 + \mathbf{v}_2 + \ldots + \mathbf{v}_n\right)$

Podle toho může učitel informatiky sestrojit počítačový model, který lze porovnat s pozorováním.

Rychlost pohybu: Fyzikové chápou křivku jako množinu bodů, kterými prochází pohybující se bod. V tomto smyslu křivku „vytváří“ bod svým pohybem. Při konstrukci křivek zadaných parametrickými rovnicemi lze tedy parametr chápat jako čas. Pro učitele informatiky by neměl být problém napsat program, který podle algoritmů popsaných v závěru kpt. 3 sestrojuje křivky zadané parametricky a křivky v polárních souřadnicích s možností tyto křivky přerušovat dle závěru kapitoly předchozí. Pak lze např. bodovou funkcí $f(t) = \left[\cos t; \sin t\right]; t \in \left\langle 0; 2\pi \right\rangle$ modelovat rovnoměrný pohyb po kružnici. Šikmý vrh v gravitačním poli Země lze rozložit do horizontálního a vertikálního směru a popisuje ho bodová funkce $h(t)=\left[v_0 t\cos\alpha; v_0 t\sin\alpha-\frac{1}{2}gt^2\right]$. Lze si tak vcelku jednoduše pořídit velmi dobrou ilustraci tohoto pohybu tak, jak demonstruje obr. 8. Vyzkoušejte si.



Obr. 8. Šikmý vrh v gravitačním poli Země

Rovnice

$$\rho=-\frac{1}{1+\varepsilon\cdot\cos t} \qquad (1)$$

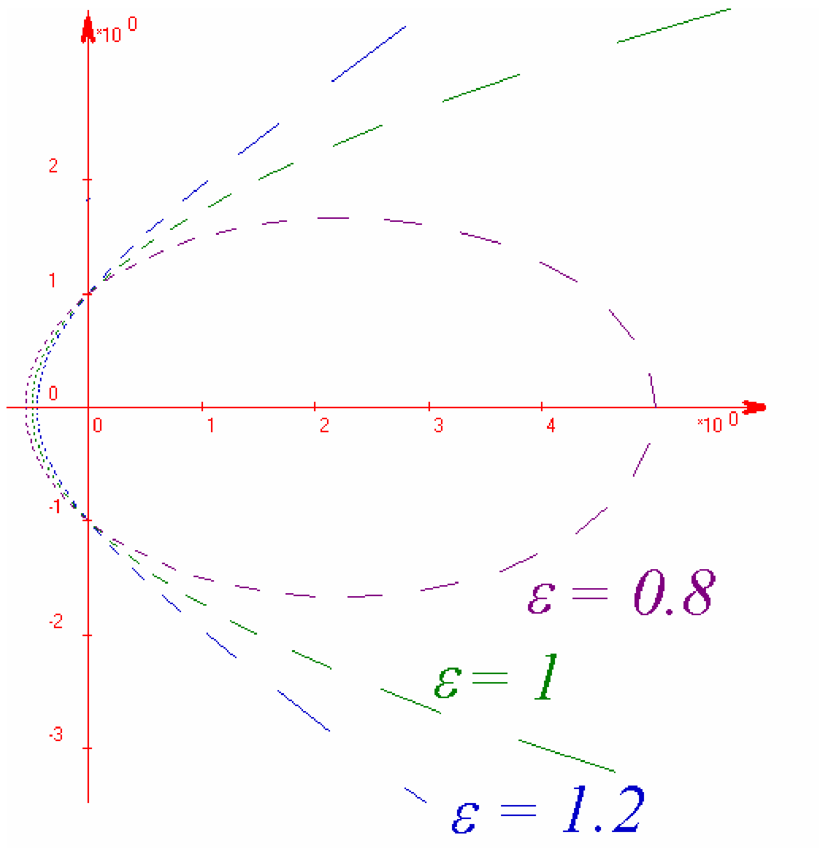
je rovnice kuželosečky v polárních souřadnicích s ohniskem v počátku ($0<\varepsilon<1$ - elipsa, $\varepsilon=1$ - parabola, $\varepsilon>1$ - hyperbola). Parametr t zde má význam úhlu. Chápeme-li tedy tuto rovnici jako rovnici pohybu, jedná se o pohyb s konstantní úhlovou rychlostí, tedy pohyb nerovnoměrný – viz obr 9. Pokud bychom chtěli tímto způsobem (zcela korektně) demonstrovat rovnoměrný pohyb po kuželosečce, museli bychom provést tzv. parametrizaci obloukem, známou z diferenciální geometrie, což je práce pro matematika:

Vektor $\dot{\mathbf{p}}$ rychlosti hmotného bodu získáme derivací vektoru polohového a požadujeme-li rovnoměrný pohyb, musí být

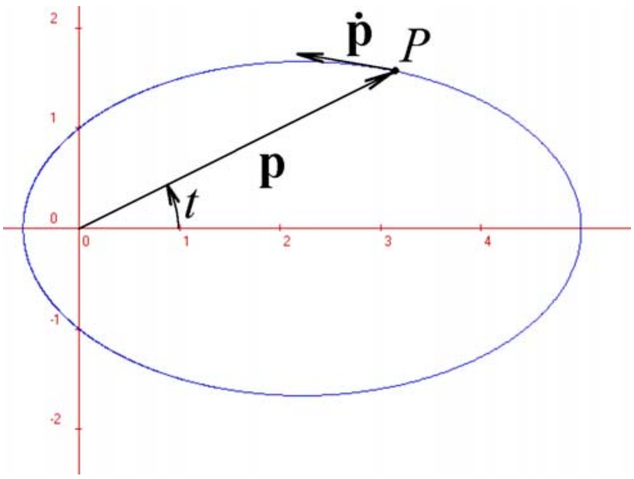
$$\|\dot{\mathbf{p}}\|=konst$$

$$\|\dot{s}(t)\|=\|\dot{\mathbf{p}}(t)\|=\sqrt{\dot{\mathbf{p}}(t)\cdot\dot{\mathbf{p}}(t)}$$

$$s(t)=\int_{t_0}^{t} v(t)\,\mathrm{d}t=\int_{t_0}^{t}\sqrt{\dot{\mathbf{p}}(t)\cdot\dot{\mathbf{p}}(t)}\;\mathrm{d}t$$



Obr. 9. Nerovnoměrný pohyb po kuželosečce (1)



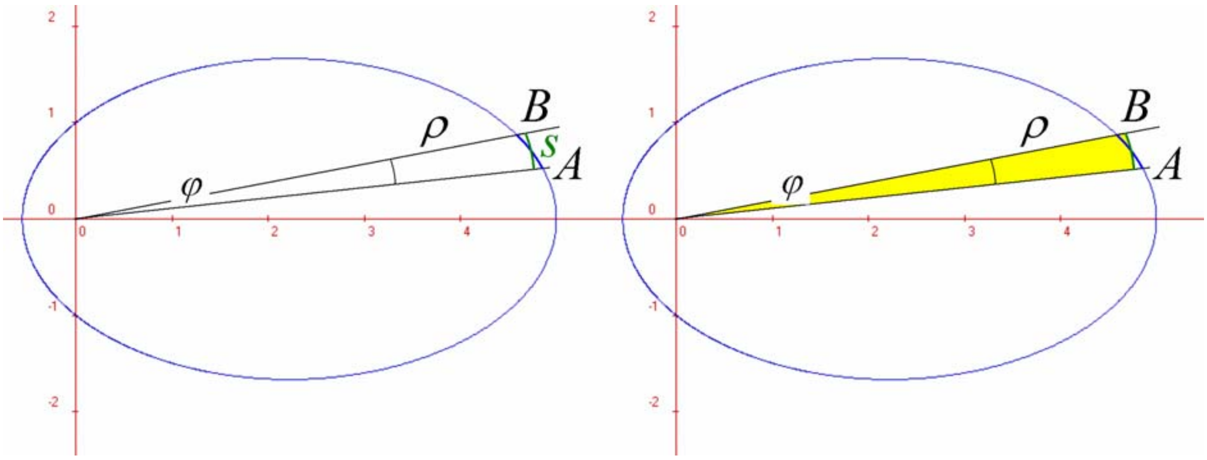
Obr. 10. Ke konstantní obvodové resp. plošné rychlosti

Protože však v našem případě je
$$p_1 = \rho \cdot \cos t = \frac{1}{1+\varepsilon \cdot \cos t} \cdot \cos t$$
$$p_2 = \rho \cdot \sin t = \frac{1}{1+\varepsilon \cdot \cos t} \cdot \sin t$$

je tento postup prakticky neschůdný. Pokud navíc chceme tímto způsobem demonstrovat pohyb planety kolem Slunce (tj. pohyb s konstantní plošnou rychlostí), museli bychom místo vektoru rychlosti $\mathbf{p}$ použít vektor plošné rychlosti $\mathbf{w} = \frac{1}{2}(\mathbf{p} \times \dot{\mathbf{p}})$ a pro dráhu pohybu konstantní plošnou rychlostí tedy obdržíme

$$S(t) = \int_{t_0}^{t} \sqrt{\mathbf{w}(t) \cdot \mathbf{w}(t)} \, \mathrm{d}t$$
$$S(t) = \frac{1}{4} \int_{t_0}^{t} \sqrt{(\mathbf{p}(t) \times \dot{\mathbf{p}}(t)) \cdot (\mathbf{p}(t) \times \dot{\mathbf{p}}(t))} \, \mathrm{d}t$$

což je cesta ještě neschůdnější. Přesto se nemusíme těchto demonstrací vzdávat. Pro školní účely zcela jistě postačí řešení přibližné. Spočívá v tom, že v původní rovnici (1) přizpůsobíme „rychlost procházení parametru".



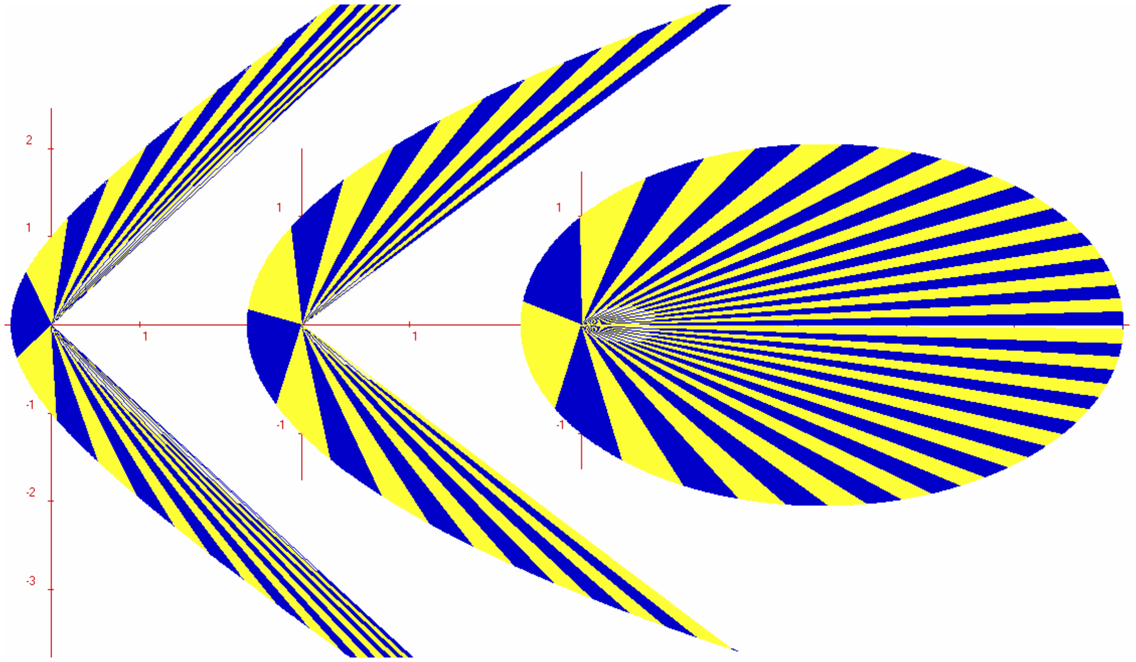
Obr. 11. K přibližnému řešení konstantní obvodové resp. plošné rychlosti

Požadujeme-li konstantní obvodovou rychlost, je třeba, aby byla konstantní délka oblouku AB (viz obr. 11 vlevo). Vzhledem k přesnosti výstupního zařízení si můžeme navíc dovolit nahradit oblouk eliptický obloukem kruhovým. Jeho délka je pak $s(t) = \varphi(t) \cdot \rho(t)$. Jestliže jsme dosud vytvářeli křivku (1) konstantním krokem parametru h_t, položme $\varphi(t) = \frac{h_t}{\rho(t)}$.

Pak bude $s(t) = h_t = konst$ a program bude generovat rovnoměrný pohyb.

Konstantní plošnou rychlost zařídíme analogicky. Eliptickou výseč (viz obr. 11 vpravo) nahradíme výsečí kruhovou, jejíž obsah je $S(t) = \frac{1}{2} \cdot \varphi(t) \cdot \rho^2(t)$. Položíme-li podobně jako v předchozím případě $\varphi(t) = \frac{2h_t}{\rho^2(t)}$, bude $S(t) = h_t$ a programem lze demonstrovat pohyb s konstantní plošnou rychlostí, a to pro všechny regulární kuželosečky (viz obr. 12.)

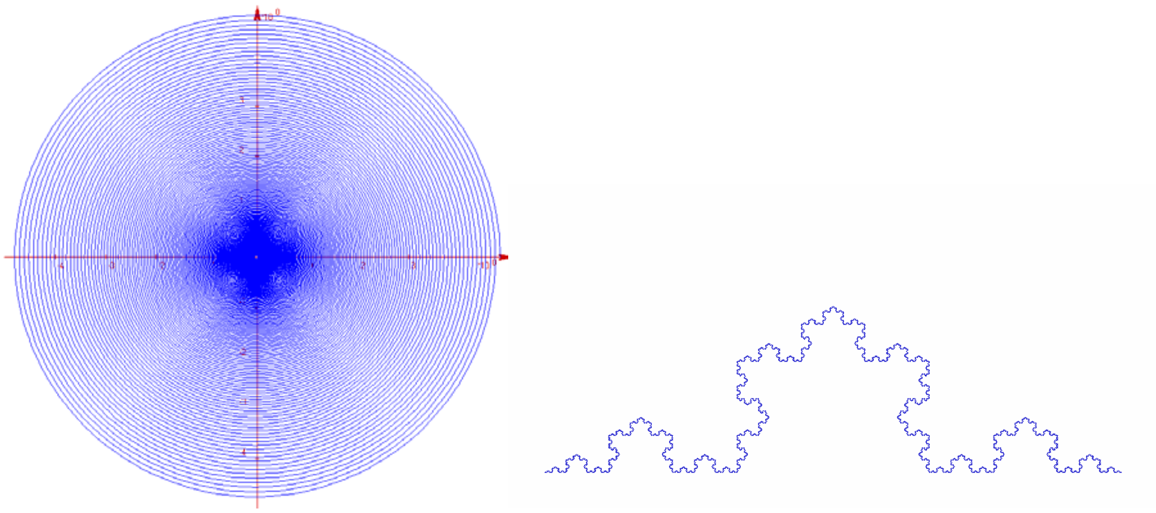
Vyzkoušejte si zde



Obr. 12. Hyperbolický, parabolický a eliptický pohyb s konstantní plošnou rychlostí

6 Může informatika pomoci matematice?

Učitel informatiky může matematikovi pomoci znázornit řadu užitečných pojmů a vlastností. I v matematice se samozřejmě hodí výše zmíněný program Rovinne_Krivky, konstrukci grafu funkce lze poměrně jednoduše doplnit o ilustraci derivace funkce. Vděčným tématem pro ilustrační programy jsou nekonečné řady. Poměrně snadno můžeme sestrojit spirálu skládající se s půlkružnic, které postupně zmenšují poloměr s kvocientem q (na obr. 13 vlevo je $q = 0{,}99$). Zajímavá je také tzv. Kochova křivka, kterou si můžeme prohlédnout na obr. 13 vpravo a jejíž konstrukce je ilustrována zde. Poměrně jednoduše lze spočítat délku těchto křivek:

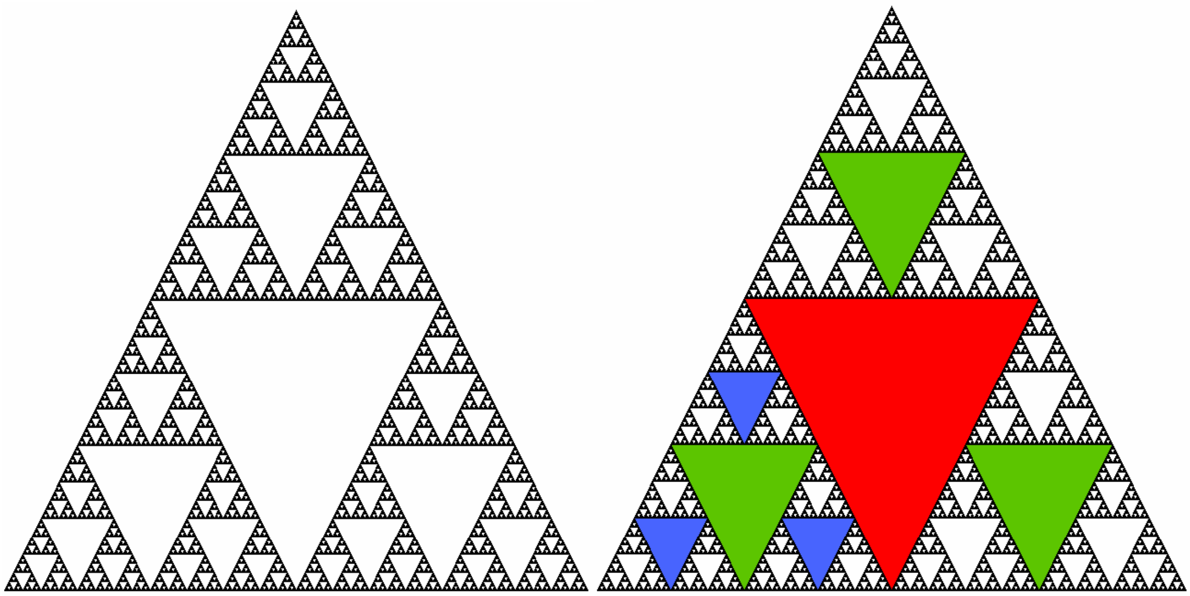


Obr. 13. Půlkružnicová spirála a Kochova křivka

Spirála: $\ell = \frac{\ell_0}{1-q} = \frac{\ell_0}{1-0.99} = 100\ell_0$ Kochova křivka: $\ell = \ell_0 \cdot \lim\limits_{n\to\infty} \frac{4^n}{3^n} = \infty$

Spirála má tedy konečnou délku, kdežto Kochova křivka je nekonečná…

Dalším zajímavým útvarem je např. Sierpiňského trojúhelník, který vznikne tak, že z trojúhelníka (nikoli nutně rovnostranného – viz obr. 14) odebíráme do nekonečna trojúhelníky sestrojené ze středních příček.

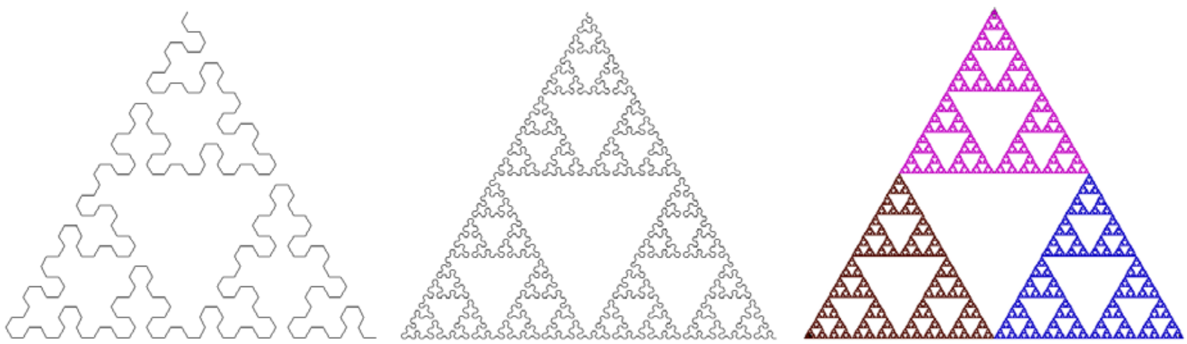


Obr. 14. Sierpinského trojúhelník jako „plocha“

Jaký je obsah tohoto útvaru? Označíme-li S_Δ obsah původního trojúhelníka a $\overline{S}$ obsah odebraných částí, máme

$$S = S_\Delta - \overline{S} = S_\Delta - \sum_{n=0}^{\infty} \frac{3^n}{4^{n+1}} S_\Delta = S_\Delta - \frac{1}{4} S_\Delta \sum_{n=0}^{\infty} \frac{3^n}{4^n} = S_\Delta - \frac{1}{4} S_\Delta \cdot 4 = 0$$

Tento útvar má tedy nulový obsah a lze ho považovat za limitní případ posloupnosti speciálních křivek, jak ilustruje obr 15 (konstrukce těchto křivek však již vyžeduje poněkud speciální znalosti teorie gramatik)



Obr. 15. Sierpinského trojúhelník jako „křivka“

7 Jak souvisí informatika s matematickými znalostmi?

Matematické znalosti mohou mít různou úroveň. Podle mého názoru lze rozlišit čtyři stupně znalosti jednoho a téhož tématu. Tento svůj názor se pokusím ilustrovat na jednoduchém příkladu – aritmetickém průměru.

1. Odříkání definice: *Aritmetický průměr spočítáme, když sečteme všechny hodnoty a vydělíme jejich počtem.*

Na této úrovni se ještě o žádnou „znalost“ nejedná. Tuto větu by zvládl možná i cvičený papoušek.

2. Počítání příkladů:

i	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*	*11*	*12*
x[i]	*6,2*	*4,5*	*2,8*	*9,1*	*2,9*	*7,5*	*4,2*	*3,7*	*8,1*	*1,6*	*5,7*	*4,9*

$$\overline{x} = 5{,}1$$

Obávám se, že na této úrovni dnes zůstává většina středoškoláků.

3. Obecný zápis:

$$\overline{x} = \frac{\sum_{i=1}^{n} x_i}{n}$$

Z obecných matematických vzorců má bohužel většina populace naprostou hrůzu. Přitom používání takových zápisů je jedním z nepřehlédnutelných znaků abstraktního myšlení.

4. Naprogramování:

```
Prumer:=0;
For i:=1 to n do
      Prumer:=Prumer+x[i];
Prumer:=Prumer/n;
```

Teprve naprogramování nějakého problému signalizuje jeho dokonalé pochopení. Prokazuje totiž schopnost „vysvětlit“ problém někomu, kdo o něm nemá (a nemůže mít) ani ponětí a kdo neodpustí ani sebemenší chybičku. Toto je však bohužel na většině středních škol značně podceňováno

Pokud v duchu předchozího textu naučíme studenty vidět v obecných matematických zápisech jen úsporné zápisy algoritmů, mohou se je naučit téměř mechanicky „překládat“ do programovacího jazyka, např.

$$S = \sum_{k=0}^{n} \boxed{výraz}$$

```
Soucet:=0;
For k:=0 to n do
      Soucet:=Soucet+{výraz};
```

$$S = \prod_{k=0}^{n} \boxed{výraz}$$

```
Soucin:=1;
For k:=0 to n do
    Soucin:=Soucin+{výraz};
```

Před studenty i učitelem se pak otevře řada dalších možností. Místo zadání typu „Sestavte program, který vypíše všechna sudá čísla od dvou do tisíce“, je možné řešit problémy typu „odhadněte $\lim\limits_{n\to\infty} s_n$, je-li

$$s_n = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \ldots + (-1)^n \cdot \frac{4}{2n+1} = 4 \cdot \sum_{k=0}^{n} \cdot \frac{(-1)^k}{2k+1}$$

$$s_n = \left(1+\tfrac{1}{n}\right)^n = \underbrace{\left(1+\tfrac{1}{n}\right)\cdot\left(1+\tfrac{1}{n}\right)\cdot\ldots\cdot\left(1+\tfrac{1}{n}\right)}_{n\times} = \prod_{k=1}^{n}\left(1+\tfrac{1}{n}\right)$$

Jsou to známá čísla a je možné třeba vypsat soutěž o to, kdo je jako první pozná. Přitom je to látka především pro informaticka, takže matematika to nestojí žádný čas a informatik na tom může učit programování cyklů třeba místo oblíbeného přehazování písmenek ve slovech, „čtení pozpátku“ apod.

8 Kolik klacků máme pod nohama?

Znají to snad všichni učitelé matematiky, fyziky a technických předmětů: Student dvakrát podtrhl jako výsledek zcela nesmyslné číslo. Pak se brání, že je to určitě dobře, protože mu to vyšlo na kalkulačce či na počítači. Většinou je to následek nějakého překlepu. Může se ovšem stát, že stejný výsledek vyjde celé třídě. Postup je v pořádku a celá třída sotva udělá stejný překlep. A celá třída je přesvědčena o tom, že je to správně. Slepou vírou v neomylnost počítače je třeba na střední škole důkladně zatřást. Počítač je totiž jako oheň – dobrý sluha, ale zlý pán.

Modelování konvergence: pojem konvergence posloupností a řad je velmi důležitý a není se co divit, že se ho snažíme studentům co nejvíce přiblížit. Jeden z velmi rafinovaných způsobů spočívá v této úvaze: Nechejme počítač sečítat. Pokud se součet přestane měnit, řadu prohlásíme za konvergentní (a hrdě to oznámíme). Pokud součet přeteče rozsah použitého typu, řada diverguje. Algoritmus můžeme popsat následovně:

```
Clen:=1;
Nasledujici_Soucet:=1;
 repeat
    Predchozi_Soucet:=Nasledujici_Soucet;
    Clen:=0.9*Clen;
    Nasledujici_Soucet:= Predchozi_Soucet+Clen;
 until Predchozi_Soucet = Nasledujici_Soucet;
```

(konkrétně se jedná o test konvergence geometrické řady s kvocientem $0{,}9$, modifikace pro řady jiné je však velmi snadná). Na první pohled se zdá, že cyklus nemůže nikdy skončit, protože podmínka, která má ukončit cyklus, nemůže být nikdy splněna. Cyklus však kupodivu skončí velmi rychle. Při použití pascalovského číselného typu real skončí program ve vteřině – po sečtení 245 členů hlásí konvergenci a součet 9,999 999 855 1 (což je výsledek velmi dobrý). Vysvětlení je velmi jednoduché: číselný typ real má k dispozici jedenáct platných číslic, přitom $s_{245} = 9{,}999\,999\,855\,051\ldots$ a $0{,}9^{246} \approx 5\cdot 10^{-12}$, tj. jak s_{245}, tak s_{246} zaokrouhleno na jedenáct platných číslic je $9{,}999\,999\,855\,1$ a hodnotu menší než $5\cdot 10^{-12}$ už počítač není schopen přičíst.

Tímto testem konvergence projde úspěšně i řada

$$s = 4 - \frac{4}{3} + \frac{4}{5} - \frac{4}{7} + \frac{4}{9} - \ldots + (-1)^n \cdot \frac{4}{2n+1} + \ldots = 4 \cdot \sum_{k=0}^{\infty} \cdot \frac{(-1)^k}{2k+1}$$

Použijeme-li v Pascalu typ real, pak si na výsledek sice počkáme několik hodin, ale je možno to zadat jako domácí úkol pro počítač, který by jinak zahálel. Počítač sečte asi dvacet miliard členů a dostaneme

$$\pi \approx 3{,}141\ 592\ 653\ 6$$

Nechceme-li na výsledek čekat tak dlouho, můžeme pracovat s typem single (sedm platných číslic). Výsledek

$$\pi \approx 3{,}141\ 592$$

dosaneme asi za minutu.

Plovoucí desetinná čárka, která pomohla odhalit konvergenci předchozích řad, je však nebezpečným „klackem pod nohama“. Jestliže jsme takto studentům řekli A, je třeba říci i závažné B a studenty důrazně varovat: nevěřte počítači! Testujeme-li totiž podobným způsobem řadu

$$r = 4 + \frac{4}{3} + \frac{4}{5} + \frac{4}{7} + \frac{4}{9} + \ldots + \frac{4}{2n+1} + \ldots = 4 \cdot \sum_{k=0}^{\infty} \frac{1}{2k+1}$$

dostaneme v typu single během pěti vteřin (!) „výsledek“ $r = 32$ (!!)

Zneužitá plovoucí desetinná čárka se krutě pomstila. Na rozdíl od aritmetiky reálných čísel je pro ni totiž konvergenční podmínka $\lim a_n = 0$ nejen podmínkou nutnou, ale rovněž podmínkou postačující. Přitom lze studentům známým srovnávacím kriteriem velmi jednoduše předvést, co nám vlastně počítač sečetl:

$$r_k = 4 \quad + \frac{4}{3} + \frac{4}{5} \quad + \frac{4}{7} + \frac{4}{9} + \ldots + \frac{4}{25} \quad + \frac{4}{27} + \frac{4}{29} + \ldots + \frac{4}{125} \quad + \frac{4}{127} + \frac{4}{129} + \ldots + \frac{4}{625} + \ldots >$$

$$4 \quad \underbrace{+ \frac{4}{5} + \frac{4}{5}}_{2\times} \quad \underbrace{+ \frac{4}{25} + \frac{4}{25} + \ldots + \frac{4}{25}}_{10\times} \quad \underbrace{+ \frac{4}{125} + \frac{4}{125} + \ldots + \frac{4}{125}}_{50\times} \quad \underbrace{+ \frac{4}{625} + \frac{4}{625} + \ldots + \frac{4}{625}}_{250\times} + \ldots =$$

$$4 \quad + \frac{8}{5} \quad + \frac{40}{25} \quad + \frac{200}{125} \quad + \frac{1000}{625} + \ldots =$$

$$4 \quad + \frac{8}{5} \quad + \frac{8}{5} \quad + \frac{8}{5} \quad + \frac{8}{5} + \ldots$$

Jak se násobí zlomky? Domníváte se, že $\frac{a}{b} \cdot \frac{c}{d} = \frac{ac}{bd}$? U počítače na to raději zapomeňte. Položte $a = b = 10^{154}$; $c = d = 2 \cdot 10^{154}$ a nechejte počítat Excel. Levou stranu zvládne zcela přesně, pravou vůbec. Pascal je v tomto směru odolnější, když mu ale přidáte do exponentu nulu, dopadne stejně.

Nechejte počítač vyčíslit výraz

$$\frac{1\cdot 3\cdot 5\cdot\ldots\cdot(2n-1)}{2\cdot 4\cdot 6\cdot\ldots\cdot(2n)}. \tag{2}$$

Něco takového stráví Excel jen pro $n \le 150$. Po protažení vzorce o buňku dál výpočet zhavaruje. Pascalovský extended vydrží sice podstatně více, ale pro $n = 1605$ „spadne“ také. Buňky generující výraz

$$\frac{1}{2}\cdot\frac{3}{4}\cdot\frac{5}{6}\cdot\ldots\cdot\frac{2n-1}{2n} \tag{3}$$

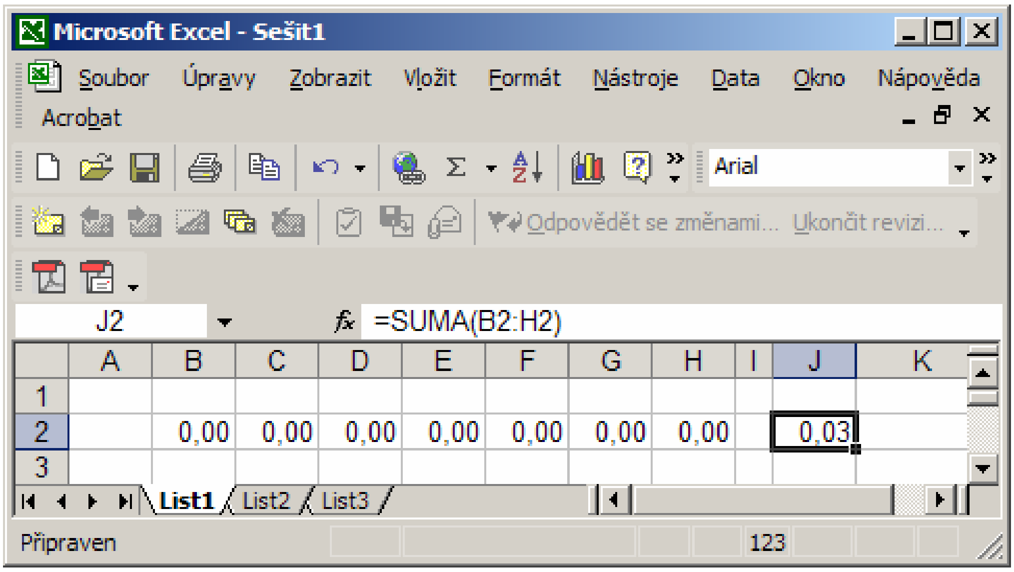
můžete v Excelu protahovat, co hrdlo ráčí, a v Pascalu n zvyšovat, jak jen libo. Hravě si s tím poradí i „nejmenší“ reálný datový typ single...

Vysvětlení je velmi jednoduché – při počítání čitatele či jmenovatele výrazu (2) jednoduše přetekl číselný typ, ve kterých ten či onen program počítá. Při počítání výrazu (3) se to prakticky stát nemůže. Přetečení číselného typu se může chovat dosti záludně (záleží na nastavení kompilátoru). Například v pascalu dostaneme pro $n = 10$ hodnotu $\left(\frac{n+1}{n}\right)^n = 2{,}581\ 174\ 792$, kdežto $\frac{(n+1)^n}{n^n} = 0{,}118\ 874\ 503$. To je rozdíl velmi podstatný, nepoučený uživatel přitom bude těžko rozhodovat, co je správně.

Je sčítání komutativní? Domníváte se, že $a+b=b+a$? I pravdu, která je zřejmá asi už každému prvňákovi, můžete u počítače zahodit. Nevěříte? Miliontý částečný součet harmonické řady v pascalovském typu single (zkusí ho někdo zjistit v Excelu?) počítaný zleva doprava je 14,357..., zprava doleva pak 14,392.... Který je správně? Ani jeden. Jde jen o to, který je pravdě blíž. V typu real se sice tyto výsledky liší až na dvanáctém desetinném místě, ale i v typu extended jsem už jsem viděl součet zcela nesmyslný. Byl zakuklený v hlubinách obrovského programu a chyba se hledala pět měsíců. Tak dlouho stál celý výzkum mezinárodního významu, protože vyvíjený program dával zcela zjevně chybné výsledky. Kvůli špatně zvolenému pořadí sečítání…

Záludná past v Excelu Znám jednoho ekonoma, který se kdysi jako čirý samouk dostal na „uživatelské cestě“ k ICT dál, než leckterý maturant. Tabulkový procesor používal zpočátku jen jako nástroj na pohodlné generování „mrtvých“ tabulek. O nějakých vzorečcích neměl ani tušení, k výpočtům sloužila kalkulačka vedle klávesnice. Pak se o vzorečcích někde doslechl a požádal mě o radu. Stačila asi desetiminutová „úvodní instruktáž“. „Živé“ tabulky ho nadchly tak, že během velmi krátké doby se veškeré finanční toky jednoho velkém podniku kompletně proměnily ve sto padesát stránek tabulek rafinovaně propletených stovkami vzorečků a vazeb (Co kdyby někdo podobný postup vyzkoušel ve škole se studenty?).

Asi za měsíc mi opět zavolal. Měl problém. V jednu chvíli se v jednom sloupci s asi šedesáti hodnotami objevila všechna čísla celá, přitom ale jejich součet byl 162 352,64. Zoufalý uživatel tři dny prohledával vzorečky ve snaze objevit, co se kam připletlo. Marně. Ani já jsem dost dlouho nechápal (s Excelem pracuji velmi zřídka). Pak jsem pochopil a v rychlosti jsem vyrobil něco, co je ilustrováno na obr. 16. Šokovaný ekonom na to vytřeštil oči a z Excelu byl vyléčen. Všechny své tabulky bez milosti „usmrtil“ a vedle klávesnice opět položil kalkulačkou… Zastánci Excelu jistě mohou namítnout, že jsem ho znechutil něčím, co šachisté nazývají „levným kavárenským trikem“. Jenže „slušný“ software by svým uživatelům takto klacky pod nohy házet neměl. Jak svědčí předchozí příklady, je jich tam už z principu počítače víc než dost.



Obr. 16: A pak že 0 + 0 = 0…

9 Kudy na třetí nástupiště?

Učme MS Office, protože to **dnes** potřebuje každý úředník!

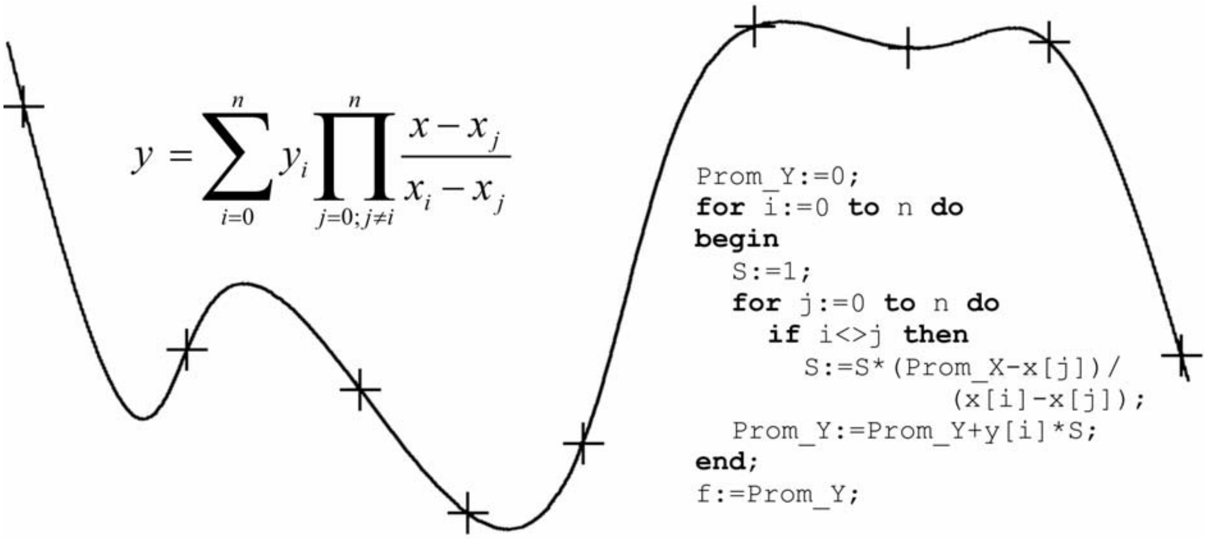
Ne, učme číselné soustavy, protože to **dnes** potřebuje každý správce sítě!

Oba tyto názory ženou vodu na mlýn cynikům, kteří se takové škole posmívají, že je jen přípravou na minulou válku. A mají bohužel pravdu. V duchu takto chápané „zásady spojení teorie s praxí“ strávila kdysi moje generace ve škole celý rok manipulací s různými typy logaritmických pravítek...

Dovolte mi proto poslední příklad. Podívejme se na vzorec na obr. 17.

Pro většinu smrtelníků něco naprosto odporného. Jestliže ovšem studenti pochopí, že je to zápis algoritmu a že odporný se zdá jen proto, že se snaží o co největší stručnost, měli by z něho mít radost. Jak jsem se snažil naznačit v kpt. 7, je překlad takového vzorečku pro počítač čistě mechanickou záležitostí. Mám to štěstí, že se studenty některých oborů na naší škole mohu s několika takovými vzorečky pracovat dřív v počítačové grafice než v numerické matematice. Zadám „neznámý“ vzorec, který člověk objevil v době, kdy elektřina sotva hýbala žabími stehýnky. O tom, že tatáž elektřina bude jednou počítat a kreslit, neměl pan Lagrange ani tušení. Nyní mají studenti za úkol „dosadit“ jeho tehdejší objev za funci `f` do procedury `Graf_Funkce` (viz kpt.3). Hodnoty x_i; y_i jsou souřadnice bodů, které má za sebou zanechávat myš, náhodně klikající po grafickém okně. Studenti nevědí, jak to má fungovat nebo co to má dělat. Pracují.

„Wow!“ ozve se náhle místo antického heuréka, když první interpolační polynom neomylně trefí všechny náhodně zadané body. Právě tato vteřina je na celé výuce nejpodstatnější. Prožíval ji už první člověk, který seskočil ze stromu. Prožívalo ji celé lidstvo, když Neil Armstrong seskočil z poslední příčky žebříku lunárního modulu. A člověk ji bude prožívat, dokud bude člověkem.



Obr. 17. I odporný vzorec vypadá jinak, když vám začne fungovat pod rukama…

S Lagrangeovým interpolačním polynomem si můžete pohrát zde.

Jeden moudrý aforismus praví, že vzdělání je to, co v člověku zůstane, až zapomene všechno, co se naučil ve škole. A já se ptám: Co zůstane v dnešních studentech, až zapomenou, v kterých že to hlubinách MS Office se nastavuje zaokrouhlování, mezera mezi odstavci a stupnice na osách spojnicového grafu? Obávám se, že v někom vůbec nic.

Učíme-li textový editor tak, že vrcholem našeho snažení je pozvánka na ples, je textový editor cílem vzdělávání. A to je špatně. Učíme-li programovat tak, že skončíme u výstupního okna posypaného hvězdičkami, je programování cílem vzdělávání. A to je špatně. Cílem výchovy truhláře přece nemůže být hoblování ani broušení hoblíku. Truhlář potřebuje vědět, jak se dělá polička.

Cílem středoškolské informatiky tedy nemohou být textové a tabulkové procesory, číselné soustavy a DNS servery. Nemůže to být ani Pascal ani C#. Nevíme totiž dne ani hodiny, kdy budou tyto nástroje dneška následovat logaritmické pravítko do propadliště dějin.

Učme **pomocí** textových editorů a tabulkových procesorů. Učme **prostřednictvím** číselných soustav a DNS serverů. Neučme programovat. Učme **programováním**. Pomocí nástrojů dneška učme schopnosti zakousnout se do problému. Učme hledat nejlepší nástroj, který je momentálně k dispozici. A učme prožívat radost z úspěchu. To je to, co vždycky hnalo, žene a bude hnát člověka dopředu. Včera, dnes, zítra i za sto let.

Středoškolská informatika dnes zmateně pobíhá mezi dvěma nástupišti a hledá svůj vlak do stanice Zítřek. Jenomže oba vlaky, mezi kterými si nemůže vybrat, jedou do pekel. Měli bychom všichni vyrazit k nástupišti číslo tři. A pospěšme si. Náš vlak do budoucnosti možná právě teď odjíždí…