INFORMATIKA

Unikátní učební osnovy informatiky pro SŠ v Izraeli

JUDITH GAL-EZER – HASHIM HABIBALLA

Computer Science Division, The Open University of Israel – Ostravská Univerzita

Informatika patří k mladým vědním disciplínám, které se i na teoretické úrovni stále dynamicky vyvíjejí a zároveň můžeme i v našem praktickém životě vidět množství jejích aplikací, kterým se nevyhne prakticky nikdo. Příprava odborníků na vysokých školách nejen pro onu teorii, ale i pro praxi, vyžaduje dobrou přípravu i na školách středních podobně jako je tomu u jiných plnoprávných předmětů-vědních disciplín jako je chemie nebo fyzika. Zároveň se nelze, stejně jako u těchto předmětů, spokojit s výukou pouze budoucích frekventantů tohoto oboru, ale je nutné dát alespoň studentům všeobecných středních škol solidní základy informatiky. U informatiky se bohužel v poslední době setkáváme s nepochopením tohoto oboru a často se kompetence ve výuce zaměňují za tzv. informační gramotnost. Informační gramotnost je jistě nezbytnou výbavou každého absolventa SŠ, na druhou stranu by nemělo být přípustné, aby se tak dělo na úkor informatiky. Můžeme dát analogii například s fyzikou, kde bychom jistě také nechtěli vytlačit klasické disciplíny jako je mechanika, optika atd., aby byl prostor studenty naučit, jak si opravit některé poruchy spalovacího motoru svého automobilu.

Ve výuce hraje velkou roli profesní osobnost učitele a jeho vlastní připravenost v klíčových oblastech informatiky (algoritmizace, teoretická informatika), která je u nás na solidní úrovni (odborná informatika je v přípravě učitelů velmi podobná přípravě odborníků). Totéž se už bohužel nedá říci o učebních osnovách informatiky a požadovaných výstupních znalostech

a dovednostech absolventa SŠ. Ty kladou zcela zásadní důraz právě na informační gramotnost a velmi malý (nebo dokonce nepovinný) prostor je věnován tématům jako je algoritmizace a rozvoj algoritmického myšlení. Nechceme zde vytvořit filozofickou úvahu na téma obsahu výuky informatiky, což by jistě bylo také užitečné, ale chceme učitelskou veřejnost v ČR seznámit s unikátními učebními osnovami, které vznikly během mnohaletého výzkumu týmu odborníků v Izraeli. Tyto osnovy a vytvořená bohatá paleta učebnic a dalších výzkumů může být zajímavou inspirací kam směřovat i českou didaktiku informatiky nebo se z ní alespoň částečně poučit (učebnice jsou pochopitelně s ohledem na studenty vytvořeny v Hebrejštině).*)

1. Výzkumný tým a jeho cíle

Komise, založená Izraelským ministerstvem školství, vytvořila v roce 1990 nové učební osnovy informatiky pro střední školy (v Izraeli odpovídá střední škola 10. – 12. ročníku školní docházky). Tato komise se skládala z informatiků z různé úrovně školství (SŠ, VŠ) a z ministerstva. Učební osnovy byly bohatě publikovány na mezinárodní úrovni [2]a jsou rovněž dostupné na Internetu: http://www.openu.ac.il/Personal_sites/download/galezer/high-school-program.pdf, http://www.openu.ac.il/Personal_sites/download/galezer/curr_and_syll.pdf

Spoluautorka tohoto článku (Prof. Gal–Ezer) byla členem komise a nyní je její předsedkyní. Učební osnovy již byly vyzkoušeny v rámci pilotního projektu na vybraných školách a nyní již byly oficiálně schváleny ministerstvem pro výuku na všech školách.

Principy, podle kterých byly osnovy tvořeny, jsou následující (a věříme, že jde o principy platné celosvětově bez ohledu na konkrétní podmínky dané země).

- *Informatika by měla být vyučována na rovnocenné úrovni* jako jiné vědní obory (fyzika, chemie, biologie).
- *Výuka by se měla koncentrovat na klíčových konceptech oboru.* Zejména je potřeba zdůraznit pojem algoritmického problému a jeho řešení –

*) Spoluautor (Habiballa) by rád vyjádřil své díky za poskytnutí bohatých materiálů a konzultace k tématu článku, které umožnily překlad materiálů do českého jazyka, a to především Prof. Gal-Ezer – předsedkyni komise pro nové učební osnovy informatiky.

algoritmu. Obsah výuky musí zahrnovat především koncepty nikoliv měnící se technologie.

- *Je potřeba vytvořit odlišné modely výuky (program s 1, 3 a 5 tématy).* Provede se diferenciace programů pro školy (studenty) s neinformatickým zaměřením a se zaměření na informatiku.
- *Každý program musí mít povinnou a volitelnou část pro vyšší flexibilitu.*
- *Je uplatněn princip „zipu" – konceptuální (teoretická) výuka je kombinována s experimentální.*
- *Měla by být vyučována dvě rozdílná paradigmata při programování.* Neměl by se tedy udržovat zastaralý způsob jednoho způsobu algoritmického myšlení a řešení problémů (většinou procedurální paradigma).
- *Je nezbytné vytvořit kvalitní učebnice pro všechna vyučovaná témata.* Tyto učebnice tvoří několik nezávislých týmů na akademických institucích, přičemž ovšem tyto týmy tvoří jednak odborní informatici, učitelé středních škol i vědci z oboru didaktiky informatiky.
- *Všichni učitelé musí nezbytně mít formální odborné vzdělání v informatice.* Toto vzdělání musí odpovídat alespoň bakalářskému ODBORNÉMU studiu informatiky (samozřejmě akreditované učitelské studijní programy v ČR toto mají splňovat).

2. Učební osnovy

Osnovy se dělí do pěti modulů, z nichž některé mají více než jednu alternativu. Každý modul má dotaci 90 hodin.

- **Základy 1 a 2:** Tento zdvojený modul (180 hodin celkem) má dát základy algoritmizace jednak teoreticky (bez konkrétního programovacího jazyka) a paralelně probíhá výuka v konkrétním programovacím jazyce (procedurálním).

  Výuka v Základech 1 zahrnuje následující témata:

  1. Úvod (koncept programu – algoritmu – a jeho vstupů a výstupů, ale také velmi stručně popis počítačů, hardware, software, operačních systémů, programovacích jazyků, kompilace a spouštění programů),
  2. základní model výpočtu (popis prvků procedurálního programování, proměnné, přiřazení, jednoduchá sekvence příkazů),
  3. návrh algoritmů pro řešení problémů (dekompozice problému na podproblémy),

4. podmíněné vykonávaní příkazů (podmínka jako nástroj algoritmizace, nejsou na této úrovni vyučovány ani vnořené operátory ani operátor negace),
5. správnost algoritmu (testování, syntaktické, run-time chyby a chyby výstupů algoritmu),
6. iterativní vykonávání příkazů (jednoduché iterace, čítače, akumulátory, ukončovací podmínka, nekonečná iterace, příkaz „while“),
7. efektivita algoritmů (identifikace a vyčíslení složitosti dominantních částí algoritmu, časová složitost jako funkce velikosti vstupu algoritmu, porovnání složitosti, nejhorší případ – v žádném případě nejde o formální definici časové složitosti),
8. funkce (rozložení na podproblémy, volání funkce, tělo funkce, volání parametrů hodnotou, lokální proměnné),
9. jednorozměrná pole (pole jako kolekce proměnných stejného typu, spojení s iterací jako prostředkem pro práci s polem),
10. složitější příklady (využití dovedností z předchozích kapitol pro procvičení, skládání jednodušších sekvencí a struktur pro řízení programu).

Výuka v Základech 2 zahrnuje následující témata:

1. Návrh algoritmů (složitější problémy, rozklad „shora-dolů“),
2. základní model výpočtu (popis prvků procedurálního programování, proměnné, přiřazení, jednoduchá sekvence příkazů),
3. procedury (rozšíření učiva z modulu Základy 1, typy volání parametrů, rozsah platnosti proměnných),
4. datové typy (deklarace typu, výčtové typy, pole),
5. rekurze (koncept, výhody a nevýhody, vztah k rekurzivní definicím funkcí – faktoriál, Fibonacciho posloupnost),
6. znaky a řetězce (práce s texty, operace s řetězci),
7. pokročilé algoritmické problémy (vyhledávání, řazení),
8. rozšíření správnosti a efektivity algoritmů (hlubší diskuze o správnosti a časové složitosti i ve spojitosti s návaznými kapitolami – rekurze).

- **Druhé paradigma a aplikace:** Uvádí studenty do druhého programovacího paradigmatu nebo přináší aplikace informatiky – např. informační systémy nebo počítačovou grafiku a to jak teoreticky tak prakticky. V případě, že obsahem je druhé paradigma, musí jít o odlišný přístup než procedurální.

Druhé paradigma se primárně zaměřuje na logické programování. Nejprve se provádí výklad výrokové a predikátové logiky, včetně rozdílů vyplývajících pro logické programování. Dále se probírají klasická témata jako při vysokoškolské výuce jazyka Prolog. Je potřeba zdůraznit, že se zde studenti seznámí s principy formální dedukce (což je nezbytné pro pochopení tohoto paradigmatu), což v naší středoškolské výuce logiky zcela chybí.

Alternativně se vyučuje architektura počítačů, včetně Assembleru (resp. jazyků nízké úrovně). Zde se vyučují číselné soustavy, architektura procesoru, strojový jazyk procesoru a následně principy assembleru, zásobníku, přerušení.

Další alternativou je počítačová grafika. Zahrnuje rozdíl mezi 2D a 3D grafikou a jejich implementace z hlediska hardwarového (buffery, vstupně-výstupní zařízení), 2D modely (bodová reprezentace, drátový model, CSG atd.), vrstvy, reprezentace křivek a ploch (polygony, B-spline, Bezierovy křivky, plochy), zobrazování, transformace, tvorba obrazu (algoritmy vykreslování – DDA, Cohen-Sutherland, atd.), standardy grafických formátů (principy, algoritmy – OPENGL, GIF, atd.). Důležitý je také projekt, který studenti musí sami naprogramovat – jedná se o funkční projekci 3D objektů.

Poslední detailně popsanou aplikační alternativou jsou informační systémy (IS). Definuje se pojem informačního systému a jeho využití, akce v IS – interaktivní vs. dávkové zpracování, dotazy a reporty, updaty, transakce. Důraz na relace a relační databáze, včetně SQL, E-R modelu, Data-Flow diagramů. Opět se zde zpracovává praktický projekt.

- **Softwarový návrh:** Má prohloubit dovednosti ze Základů 1, 2 (důraz na datové struktury, abstraktní datové typy) a navázat integrací do větších softwarových celků. Zahrnuje témata jako práce s knihovnami funkcí, abstraktní datové typy, zásobník, seznam, časová a prostorová složitost včetně O-notace, třídy složitosti, binární strom, případová studie (návrh „komplexního softwarového balíku“ – příprava na praxi).
- **Teorie:** Tento poměrně exotický modul má dát studentům náhled na složité (ale fundamentální) partie teoretické informatiky. Buď je kompletně tvořen modulem Výpočetních modelů – různé typy automatů [5]nebo tvoří Výpočetní modely jen polovinu modulu (45 hodin) a druhá část je věnována numerickým metodám.

  V modulu výpočetních modelů se zaměřuje výuka především na koncept automatu – akceptoru jazyka (nikoliv tedy na generativní koncept

gramatiky, regulárního výrazu). Je to vcelku pochopitelné, neboť automat je „selskému“ uvažování bližší a dá se uvádět na jednodušších příkladech než gramatické struktury. Výuka zabírá celou Chomského hierarchii (tedy konečné automaty, zásobníkové automaty a Turingovy stroje). Velký důraz se také klade na pochopení vztahu determinismu-nedeterminismu.

Volitelně se také vyučuje numerická matematika, což je opět zajímavá a složitá problematika, která se dá ovšem uplatnit pro řešení mnoha problémů v matematice a informatice (vždyť řešení rovnic nebo soustav lineárních rovnic je univerzální úlohou). V tomto modulu se vyučují obecné koncepty počítačových výpočtů (tedy reprezentace číselných hodnot, nepřesnosti, zaokrouhlování, kumulace numerických chyb), řešení soustav lineárních rovnic (Gaussova eliminace), řešení nelineárních rovnic (iterativní metody – bisekce, Newtonova metoda).

Z těchto modulů pak mohou být vytvořeny tři studijní verze. Verze „nízké úrovně – 1 modul“ (90 hodin), verze „střední úrovně – 3 moduly“ (270 hodin) a verze „vysoké úrovně – 5 modulů“ (450 hodin).

Verze „1 modul“ obsahuje pouze Základy 1 a je povinná v rámci těchto osnov (je vyučována buď v 10. nebo až 11. ročníku)*).

Verze „3 moduly“ obsahuje Základy 1,2 a dále modul Druhé paradigma nebo aplikace. Tento modul se rovněž vyučuje v 10. nebo 11. ročníku a to intenzivně během jednoho roku.

Verze „5 modulů“ obsahuje všechny moduly a je určena primárně pro studenty, kteří se zajímají specificky o informatiku (kterých je dnes ale samozřejmě mnoho). I když teoretický modul se může zdát pro středoškolskou informatiku jako příliš složitý, má pro potenciální studenty informatiky velký význam (záleží na způsobu podání, které nemůže kopírovat styl VŠ výuky).

3. Implementace na školách, problémy a zajímavé experimentální výsledky

Původní vzorek 8 škol (1991), kde se osnovy zkoušely, byl brzy (1994) rozšířen na 40 škol a v roce 2000 již byly osnovy zařazeny na všechny střední školy. Velkým problémem byla nedostatečná formální příprava učitelů, což se řešilo pomocí nouzových intenzivních kurzů na univerzitách,

*) Srovnejme situaci v ČR, kde výuka algoritmizace není povinná v rámci osnov a už vůbec ne v dotaci 90 hodin!

kde učitelé bez formálního vzdělání v informatice získali potřebnou kvalifikaci. To je podobný problém i v ČR, kde se také do rozšiřujících studií informatiky – obvykle alespoň tříletých – hlásí velmi mnoho již zkušených učitelů jiných aprobací než informatiky (v kontrastu s kriticky prázdnými ročníky klasického prezenčního studia učitelství). Nicméně i pro plně aprobované učitele informatiky bylo potřebné uspořádat kurzy pro seznámení s novými – revolučními – osnovami a vzniklými učebnicemi.

V rámci zavádění osnov byly mimo jiné provedeny dva zajímavé pedagogické výzkumy, jejichž závěry by pro (učitele) informatiky měly být zajímavé. První z nich se týká časové složitosti (náročnosti) algoritmů. Pokud se budeme snažit představit něco typického pro informatiku, asi nás především napadne pojem algoritmus jako řešení nějakého problému (např. seřazení posloupnosti čísel). Hned na to nás asi napadne také pojem časové složitosti [4], protože algoritmické řešení musí být také dostatečně efektivní (rychlé), aby mělo pro praxi smysl. V rámci modulu Základy 1 je zařazena také malá část věnovaná časové složitosti algoritmů (připomeňme, že tento modul je povinný). I když jde samozřejmě o dost kontroverzní téma pro středoškolskou výuku – její zařazení není nijak podobné složitosti didakticky pojaté na vysokoškolské úrovni (tedy žádné formální definice, ale pouze výpočet konkrétní složitosti – počtu elementárních kroků). Jde spíše o snahu studentům alespoň částečně umožnit pochopit, že programy řešící stejný problém, mohou být různě efektivní. Při praktické výuce byly zjištěny zajímavé časté chyby, kterých se studenti dopouštějí při chápání, zda jeden program je „rychlejší“ než druhý [3]. Tyto chyby se dají také charakterizovat pomocí univerzálních typů pro vzdělávání v oblasti přírodních věd. Byly identifikovány u vybraných frekventantů SŠ podle nových osnov.

- *Kratší program má menší časovou složitost:* Tato chyba odpovídá obecnému chybovému konceptu „čím více A, tím více B“. Studenti si myslí, že méně příkazů v programu znamená také vyšší efektivitu.
- *Méně proměnných znamená menší časovou složitost:* Podobně jako u předchozí chyby studenti spojují nesprávně počet proměnných s efektivitou.
- *Programy obsahující stejné příkazy v různém pořadí jsou stejně efektivní:* Jde o instanci obecného principu „stejné A, stejné B“.
- *Dva algoritmy provádějící stejný úkol mají stejnou složitost*

Dalším velmi zajímavým výzkumem je studie věnovaná asi nejkontroverznějšímu modulu – Teorii. Díky experimentu s vybranými studenty

se podařilo odhalit problémy ve vnímání nedeterminismu versus determinismu při návrhu konečných automatů [1]. Tento rozpor mohou vnímat i vysokoškolští učitelé při výuce teorie formálních jazyků a automatů. Nedeterminismus je pro studenty navyklé na algoritmické – tedy deterministické myšlení poněkud složité téma. Rovněž současné počítače jsou typicky deterministické stroje a proto se zdá, že nedeterminismus při návrhu automatů nutí studenta vlastně dělat přesný opak toho, co je mu během studia informatiky vštěpováno. Nemusíme při něm promýšlet odezvy na všechny možné situace (to odpovídá automatu, který nemusí být typicky nedeterministický, ale nemá ošetřeny všechny situace) a navíc můžeme v případě několika možných řešení jedné situace využít právě tuto nedeterministickou možnost (to by u deterministických strojů bylo něco nemožného). Návrh nedeterministického automatu je tedy obecně mnohem jednodušší (pohodlnější). Na druhou stranu to vyžaduje se částečně oprostit od klasického algoritmického myšlení, protože na výpočet nedeterministického automatu se lze v každém kroku dívat jako na nalezení jedné z cest, která vede k úspěšnému rozpoznání slova (automat jakoby uměl „uhodnout“, kterou možnost z několika použít).

V experimentu měli studenti možnost si sami vybrat, zda zapíší automat nedeterministicky (jednoduše) nebo si řešení zkomplikují a budou se snažit vytvořit automat deterministický (nebo částečně deterministický). Výzkum ukázal, že přibližně polovina studentů volí řešení spíše deterministické (někdy jen s lokálním nedeterminismem). Také se ukázalo, že existují velké rozdíly mezi skupinami studentů podle učitele (tedy hodně záleží na tom, jak sám učitel dokáže význam vztahu determinismus-nedeterminismus posoudit a pochopit). Ze subjektivních hodnocení vyplynuly například přímo obavy z nedeterminismu a dokonce i u jednoho z učitelů.

4. Závěry a možná inspirace pro výuku v ČR

Na poměrně malém prostoru jsme se snažili provést překlad, kompilaci a komentář k osnovám, které již jsou v Izraeli standardizovány. Mezi pozitiva, která bychom mohli částečně přejmout i v ČR patří především důraz na principy nikoliv konkrétní technologie – produkty. Bohužel do naší výuky INFORMATIKY stále více pronikají na úkor principů právě tyto rychle umírající technologie či dokonce konkrétní produkty. Místo principů počítačem podporované sazby textu učíme MS-Word. Místo konceptů počítačové grafiky učíme práci s konkrétním produktem. Podíváme-li se na modul Počítačové grafiky, právě tyto silné a hlavně přetrvávající principy

tvoří páteř výuky. Naučíme-li studenty práci s produktem, na první pohled jim tím dáme velkou výhodu, neboť bez dalšího studia jsou schopni se v praxi uplatnit. Bohužel jen na velmi omezenou dobu, protože tyto produkty rychle vystřídá jiná technologie (vzpomeňme například na výuku psaní textů v editoru T602 – 10 let zpět – což je dovednost v dnešní době nepoužitelná).

Druhou zajímavou vlastností osnov je povinná výuka algoritmizace alespoň na minimální úrovni. Jistě se najde mnoho oponentů takového přístupu, přesto se těžko dá nazývat předmět INFORMATIKA bez výuky algoritmizace alespoň na minimální úrovni (algoritmické myšlení a řešení problémů je asi nejtypičtějším znakem tohoto oboru, který ho diferencuje od jiných věd). Odvážné je rovněž zavést (ovšem na intuitivní úrovni) pojem efektivity (složitosti), přesto je to další klíčové vlastnost, která pro nasazení produktů informatiky v každodenním životě hraje nezbytnou roli.

Rovněž vybrané aplikace (počítačová grafika, informační systémy) a jejich vysoká teoretická úroveň mohou být vzorem i pro české osnovy. Absolventi s takovýmito znalostmi mohou být nasazeni při praktickém vývoji databází, programování informačních systémů na základě návrhu podle rozšířených modelovacích nástrojů a standardů.

Posledním sice asi nejkontroverznějším, ale přesto potřebným modulem je teorie. V MFI se čtenář již mohl seznámit s některými tématy, jež jsou i součástí modulu. Pochopit například vztah jazyka-gramatiky-automatu nebo determinismu a nedeterminismu je obtížné i pro studenta VŠ. Přesto by studenti uvažující o studiu informatiky měli mít možnost získat představu o těchto tématech již na SŠ (samozřejmě zcela odlišným způsobem – tedy intuitivně, na příkladech a ne formálně algebraicky). Studenti VŠ programů v informatice nemají ve většině případů zažito, že informatika je disciplína vysoce provázaná s matematikou (algebrou). Jejich představa o VŠ studiu informatiky je v lepším případě spojena s programováním a bohužel často ani to neplatí a představa je spojena se skládáním hardwarových komponent, budováním počítačových sítí a ve vůbec nejhorším případě pak s uživatelskou prací v aplikačních programech.

Kromě výše uvedených modulů, které jsou dobře použitelné i pro naši výuku, je vidět, že cíle výuky INFORMATIKY a tzv. INFORMAČNÍ GRAMOTNOSTI jsou relativně neslučitelné a bylo by zřejmě dobré je vyučovat pod samostatnými předměty. Jednak jsou cílové skupiny výuky INFORMATIKY odlišné podle zaměření studenta (viz odstupňované zapo-

jení počtu modulů) zatímco u INFORMAČNÍ GRAMOTNOSTI jde spíše o homogenní výuku pro všechny. A v druhé řadě jde v INFORMATICE o výuku plnoprávného vědního oboru, zatímco u gramotnosti jde o společensky vyžadované dovednosti absolventa (podobně jako u výuky čtení a psaní) – samozřejmě by se ideálně tyto dovednosti měli spíše vyžadovat již od absolventa základního školství.

Literatura

[1] *Armoni, M. – Gal-Ezer, J.:* On the Achievements of High School Students Studying Computational Models. In Proceedings of ITICSE, 2004, Leeds, UK.

[2] *Gal-Ezer, J. – Harel, D.:* Curriculum and course syllabi for a high-school program in computer science. Comp. Sci. Education, 2/1999(9), p.114-147.

[3] *Gal-Ezer, J. – Zur, E.:* The Efficiency of Algorithms – Misconceptions. Computers and Education, 2004, 42, 3, pp. 215-226.

[4] *Habiballa, H. – Kmeť, T.:* Vyčíslitelnost a složitost. MFI 15 (2005 – 06), č. 2 a 3.

[5] *Habiballa, H. – Vojkovský, P.:* Formální jazyky a automaty. MFI 13 (2003 – 04), č. 5 a 6.

Posloupnosti a řady v Excelu

VÁCLAV MATYÁŠ

Purkyňovo gymnázium, Strážnice

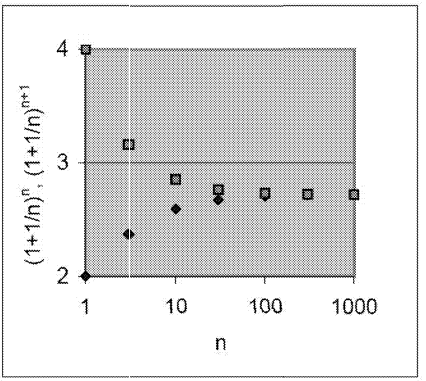
Tabulky a grafy Excelu umožňují zajímavě ilustrovat studentům základní poznatky o posloupnostech a řadách. V případech, kdy má ukázka průběhu posloupnosti zahrnout velmi mnoho členů, se ukazuje jako výhodné použít na ose x logaritmické měřítko (viz [5]).

Posloupnosti

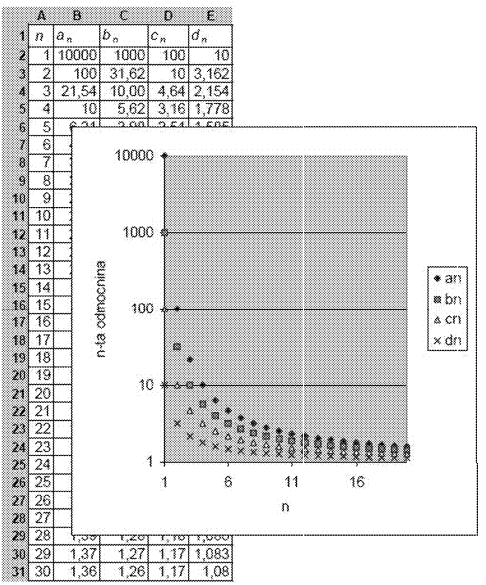
Posloupnost $\{a_n\}_{n=1}^{+\infty}$, je funkcí definovanou na množině $\mathbb{N}$ všech přirozených čísel. Pro výpočet i zakreslení limity $\lim\limits_{n\to\infty} a_n$ je v některých případech vhodné užít funkce $f(x)$ definované na množině reálných čísel. Je-li totiž $f(x)$ taková funkce, že $f(n) = a_n$ pro všechna přirozená n, pak z existence

limity $\lim_{x\to\infty} f(x) = A$ plyne, že platí $\lim_{n\to\infty} a_n = A$. Můžeme tak využít všech poznatků o ilustraci limitního procesu limit funkce $f(x)$ v nevlastním bodě $+\infty$.

	A	B	C
1	n	$(1+1/n)^n$	$(1+1/n)^{n+1}$
2	1	2	4
3	3	2,3704	3,1605
4	10	2,5937	2,8531
5	30	2,6743	2,7635
6	100	2,7048	2,7319
7	300	2,7138	2,7228
8	1000	2,7169	2,7196



Obr. 1



	A	B	C	D	E
1	n	a_n	b_n	c_n	d_n
2	1	10000	1000	100	10
3	2	100	31,62	10	3,162
4	3	21,54	10,00	4,64	2,154
5	4	10	5,62	3,16	1,778
29	28	1,39	1,26	1,16	1,080
30	29	1,37	1,27	1,17	1,083
31	30	1,36	1,26	1,17	1,08

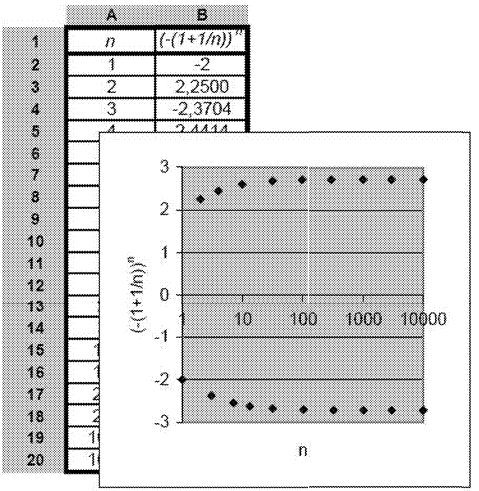
Obr. 2

Obrázek 1 pomocí tabulky a grafu ilustruje posloupnosti $\{a_n\}$, $a_n = \left(1+\frac{1}{n}\right)^n$ a $\{b_n\}$, $b_n = \left(1+\frac{1}{n}\right)^{n+1}$ pro $n = 1, \ldots, 1\,000$; první z nich je rostoucí a druhá klesající a obě mají stejnou limitu e. Stejně jako nezávisle proměnné x u spojitých funkcí, jsou indexy n vybraných členů posloupností vynášeny v logaritmickém měřítku, jenom při volbách v nabídce grafu je zadáno *nespojovat vynášené body*. Body grafu jsou pro přehlednost vyneseny jen pro některá n $(1, 3, 10, 30, 100, 300, 1000)$. Ve sloupci A tabulky jsou zadány vybrané indexy, do buňky B2 je počítáno podle vzorce =(1+1/A2)^A2. Další hodnoty ve sloupci B dostaneme tažením za pravý dolní roh buňky B2. Podobně jsou počítány hodnoty ve sloupci C.

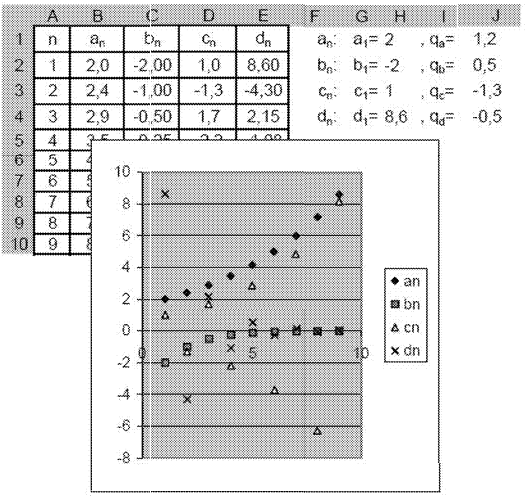
Na obrázku 2 je ilustrována věta: Nechť pro posloupnosti $\{a_n\}$, $\{b_n\}$ platí: $\lim\limits_{n\to\infty} a_n = A$, $\lim\limits_{n\to\infty} b_n = B$. Platí-li pro každé n nerovnost $a_n < b_n$, pak $A \leq B$ (tj. pro limity může nastat rovnost). Obrázek je vykreslen pro čtyři posloupnosti, jejichž n-té členy jsou: $a_n = \sqrt[n]{10000}$, $b_n = \sqrt[n]{1000}$, $c_n = \sqrt[n]{100}$, $d_n = \sqrt[n]{10}$, jak je z obrázku patrno pro $n = 1$. Pro všechna čísla p zadaná pod odmocninami zřejmě platí podle věty o limitě složené funkce: $\lim\limits_{n\to\infty} \sqrt[n]{p} = \lim\limits_{n\to\infty} p^{1/n} = p^{\lim\limits_{n\to\infty} 1/n} = p^0 = 1$. Ve sloupci A tabulky jsou zadány indexy členů posloupností; hodnota prvního členu posloupnosti $\{a_n\}$, tj. $a_1 = 10\,000$ je zadána do buňky B2 podle vzorce =100000^(1/A2) a další hodnoty ve sloupci B dostaneme tažením. Podobně jsou vypočítávány hodnoty ve sloupcích C, D, E.

Hromadný bod posloupnosti (částečná limita) může být vysvětlen na průběhu posloupnosti $\left\{(-1)^n\left(1+\frac{1}{n}\right)^n\right\}$; hodnoty členů této posloupnosti jsou vypočteny a vykresleny na obrázku 3. Tato posloupnost zřejmě nemá limitu, ale má dvě částečné limity, a to $-$ e, e. První člen posloupnosti vypočteme do B2 podle vzorce =(-1-1/A2)^A2, do buňky A2 jsme zadali počáteční index 1 a ve sloupci A je zadána posloupnost vybraných indexů těch členů posloupnosti, které chceme vykreslit (pro přehlednost opět nejsou zadány všechny). Tažením za pravý dolní roh buňky B2 pak dostáváme další členy posloupnosti.

Obrázek 4 je vykreslen k ilustraci vlastností geometrických posloupností. Zatímco u aritmetických posloupností je a_n lineární funkcí proměnné n, jejíž definiční obor je množina přirozených čísel, a vlastnosti aritmetické posloupnosti jsou přehledně určeny hodnotou diference, u geometrických posloupností záleží na znaménku prvního členu a zároveň na hodnotě kvocientu q. Z obrázku je patrné, že členy posloupností $\{b_n\}$, $\{d_n\}$ s kvocienty, pro něž platí $|q| < 1$, se s rostoucím indexem blíží k nule.



Obr. 3



Obr. 4

Změnou hodnot buněk ve sloupci J tabulky, mohou studenti velmi rychle pozorovat změny vlastností posloupností pro různé hodnoty $|q| < 1$, $q = 1$,

$q = -1$, $|q| > 1$; dále mohou ověřit (změnou znaménka hodnot ve sloupci H), že pouze změnou znaménka prvního členu posloupnosti dostaneme graf symetrický k původnímu podle osy x. Zároveň je možné připomenout vlastnosti exponenciálních funkcí.

Aritmetická řada

Připomeňme si vzorec pro n-tý částečný součet s_n aritmetické řady. Platí:

$$s_n = a_1 + (a_1 + d) + \ \ldots \ + [a_1 + (n-2)d] + [a_1 + (n-1)d],$$

$$s_n = [a_1 + (n-1)d] + [a_1 + (n-2)d] + \ \ldots \ + (a_1 + d) + a_1$$

Sečtením dostaneme

$$2s_n = n[2a_1 + (n-1)d],$$

odkud

$$s_n = \frac{d}{2} \cdot n^2 + (a_1 - \frac{d}{2}) \cdot n. \tag{1}$$

Z (1) je vidět, že $\{s_n\}$, tedy i aritmetická řada $\sum_{n=1}^{\infty} [a_1 + (n-1)d]$ je divergentní, je-li d různé od nuly. Pro $d > 0$ je $\lim_{n\to\infty} s_n = +\infty$, pro $d < 0$ je $\lim_{n\to\infty} s_n = -\infty$. Body o souřadnicích $[n; s_n]$ leží na části paraboly, jejíž vrchol je minimem křivky pro $d > 0$ nebo maximem pro $d < 0$.

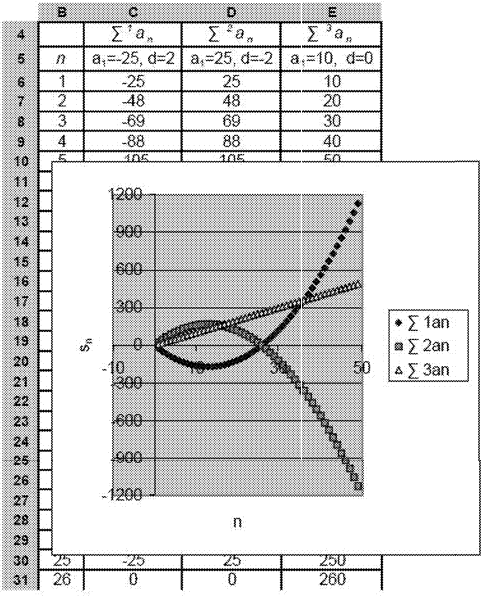
Jestliže ve vztahu (1) zaměníme n za spojitou proměnnou x, dostaneme rovnici spojité kvadratické funkce, kterou lze upravit na tvar

$$y = \frac{d}{2}\left[\left(x + \frac{2a_1 - d}{2d}\right)^2 - \left(\frac{2a_1 - d}{2d}\right)^2\right]$$

a jejímž grafem je parabola. Vrcholem této paraboly je bod $[x_0, y_0]$, kde $x_0 = -\frac{2a_1-d}{2d}$, $y_0 = -\frac{d}{2}\left(\frac{2a_1-d}{2d}\right)^2$. Může tedy nastat případ, kdy součet aritmetické řady od $n = 1$ po určité n_x roste a pro $n > n_x$ klesá, nebo naopak. Chceme-li například, aby taková změna nastala právě po desátém členu posloupnosti $\{s_n\}$, položíme

$$n_x = x_0 = -\frac{2a_1 - d}{2d} = 10,$$

takže stačí vzít $a_1 = -9{,}5\,d$. V tabulce obrázku 5 jsme pro aritmetickou řadu $\sum^1 a_n$ zvolili $a_1 = -25$, $d = 2$ (pak $x_0 = {}^1n_x = 13$, $y_0 = {}^1s_{n_x} = -169$), body $[{}^1n, {}^1s_n]$ leží na parabole s vrcholem v minimu, je zobrazeno 49 členů. Pro aritmetickou řadu $\sum^2 a_n$ jsme zadali $a_1 = 25$, $d = -2$ (pak $x_0 = {}^2n_x = 13$, ${}^2s_n = 169$), body $[{}^2n, {}^2s_n]$ leží na parabole s vrcholem v maximu, je opět zobrazeno 49 členů. Na obrázku jsou vyneseny také body $[{}^3n, {}^3s_n]$, které leží na přímce, protože příslušná aritmetická řada je zadána pro $a_1 = 10$, $d = 0$.

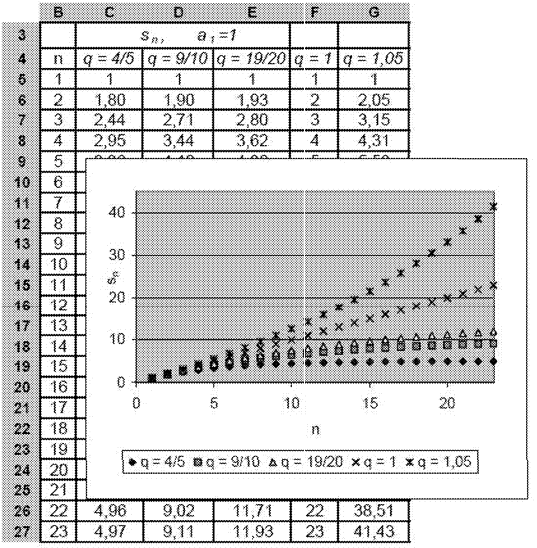


Obr. 5

V tabulce obrázku 5 jsou pro $n = 1$ zadány první částečné součty $s_1 = a_1$ jako konstanty a další částečné součty s_n jsou počítány ze vztahu $s_n = s_{n-1} + a_1 + (n-1) \cdot d$. Např. do buňky C7 je tak zadán vzorec =C6+C$6+(B7-1)*2 a tažením za pravý dolní roh buňky C7 dostaneme další hodnoty ve sloupci C. Poznamenejme, že pro výpočet částečných součtů s_n by bylo možno užít i vztah (1) a počítat je v závislosti na zadané diferenci d, počátečním a_1 a na indexu n ve sloupci B.

Geometrická řada

Geometrická řada $\sum_{n=1}^{\infty} a_1 \cdot q^{n-1}$ konverguje pro $|q| < 1$ a její součet pak je $s = \frac{a_1}{1-q}$. Na obrázku 6 jsou vyneseny částečné součty s_n pěti geometrických řad, jejichž první částečný součet $s_1 = a_1 = 1$ a které mají různé kladné kvocienty q (4/5, 9/10, 19/20, 1 a 1,05). Z obrázku je patrné, že částečné součty prvních tří konvergentních řad se blíží limitní hodnotě (5, 10, 20) tím rychleji, čím je jejich kvocient bližší nule. Pro divergentní řadu s $q = 1$ platí $s_n = n \cdot a_1 = n$. Pro divergentní řadu s kvocientem i jen o málo větším než 1 ($q = 1{,}05$) velikost s_n v závislosti na n pro větší n strmě roste, takže v tomto případě můžeme z tabulek Excelu zjistit, že $s_{100} = 2\,610$, $s_{200} = 345\,835$ a $s_{300} = 45\,479\,903$ (hodnoty s_n jsou zaokrouhleny na celá čísla).

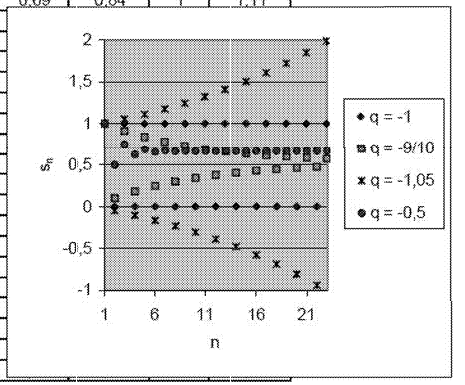


	B	C	D	E	F	G
3		s_n, $a_1=1$				
4	n	q = 4/5	q = 9/10	q = 19/20	q = 1	q = 1,05
5	1	1	1	1	1	1
6	2	1,80	1,90	1,93	2	2,05
7	3	2,44	2,71	2,80	3	3,15
8	4	2,95	3,44	3,62	4	4,31
9	5					
10	6					
11	7					
12	8					
13	9					
14	10					
15	11					
16	12					
17	13					
18	14					
19	15					
20	16					
21	17					
22	18					
23	19					
24	20					
25	21					
26	22	4,96	9,02	11,71	22	38,51
27	23	4,97	9,11	11,93	23	41,43

Obr. 6

Hodnoty $s_1 = a_1$ jsou v tabulce obrázku 6 zadány jako čísla, v dalších řádcích jsou částečné součty s_n počítány podle vztahu $s_n = s_{n-1} + a_1 \cdot q^{n-1}$, takže například do buňky C6 (řada s $q = 4/5$) je hodnota vypočítána podle vzorce =C5+C$5*(4/5)^ (B6-1) a další hodnoty ve sloupci C získáme tažením za pravý dolní roh buňky C6.

	B	C	D	E	F
3			s_n, $a_1=1$		
4	n	q = -0,5	q = -9/10	q = -1	q = -1,05
5	1	1	1	1	1
6	2	0,50	0,10	0	-0,05
7	3	0,75	0,91	1	1,05
8	4	0,63	0,18	0	-0,11
9	5	0,69	0,84	1	1,11
10	6				
11	7				
12	8				
13	9				
14	10				
15	11				
16	12				
17	13				
18	14				
19	15				
20	16				
21	17				
22	18				
23	19				
24	20				
25	21				
26	22				
27	23				



Obr. 7

Na obrázku 7 jsou vykresleny částečné součty s_n čtyř geometrických řad, jejichž první částečný součet $s_1 = a_1 = 1$ a které mají různé záporné kvocienty $q(-0{,}5; -9/10; -1$ a $-1{,}05)$. Z obrázku je patrné, že také částečné součty prvních dvou řad konvergují k limitní hodnotě s $(\frac{2}{3}, \frac{10}{19})$ a konvergují tím rychleji, čím je kvocient q bližší nule. Pro $q = -1$ nabývají členy posloupnosti $\{s_n\}$ pouze dvou hodnot 0 a 1 a pro $q = -1{,}05$ má posloupnost zřejmě dvě částečné limity $+\infty$ a $-\infty$. Částečné součty v tabulce jsou počítány podle vzorce $s_n = s_{n-1} + a_1 \cdot q^{n-1}$, přičemž hodnoty $s_1 = a_1 = 1$ jsou zadány, podobně jako v tabulce obrázku 6.

(*Pokračování*)

Literatura

[1] *Rektorys, K. a kol.:* Přehled užité matematiky. Praha, Prometheus 2003.

[2] *Odvárko, O.:* Matematika pro gymnázia. Posloupnosti a řady. Praha, Prometheus 1996.

[3] *Odvárko, O.:* Matematika pro gymnázia. Funkce. Praha, Prometheus 1996.

[4] *Stroud, K. A.:* „Engineering Mathematics", fourth ed., 1995, Macmillan Press Ltd., Hondwills, Basingstone, Hempshire and London.

[5] *Matyáš, V.:* Užití Excelu při znázorňování limit funkcí. MFI 15 (2005/6), č. 9, s. 548.