ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE
Institut vzdělávání a poradenství

Informační a komunikační technologie a jejich využití ve výuce předmětu práce s počítačem

Závěrečná písemná práce

Vedoucí: Ing. Jiří Husa, CSc.

Autor: Ing. Ondřej Hruška

Prohlášení

Prohlašuji, že jsem závěrečnou písemnou práci na téma *Informační a komunikační technologie a jejich využití ve výuce předmětu práce s počítačem* vypracoval samostatně a použil jen pramenů, které cituji a uvádím v seznamu použité literatury.

V Praze dne 2. 4. 2007

podpis autora závěrečné písemné práce .....

Poděkování

Úvodem bych rád poděkoval za vedení práce a odborné rady Ing. Jiřímu Husovi, CSc. a také Ing. Petrovi Štorkovi za odzkoušení kurzu ve výuce.

Obsah

Přílohy:

- MATURITNÍ POŽADAVKY (specifické cíle) ke zkoušce z informačně technologického základu zadávané MŠMT
- Náhled stran kurzu
- Dotazník ke kurzu

1 Úvod

Informační a komunikační technologie se stále více prosazují do mnoha různých oblastí lidské činnosti. Dnes již nikoho příliš nepřekvapí například využití počítačů v průmyslu, kde onen „počítač jako rychlý idiot“ vykonává stále se opakující činnosti. Stejně tak výpočetní techniku využívá stále větší skupina lidí jako svého pomocníka v zaměstnání i v soukromí pro běžné činnosti – od vedení osobní korespondence, získávání informací z internetu až po řízení nukleárních reakcí. O moderní společnosti se tak často mluví jako o společnosti informační.

Využití takových technologií ve vzdělávání je pochopitelným pokračováním těchto trendů. Tato oblast se obvykle nazývá e-learning (elektronické učení/vzdělávání). Zajímavé je, že se v reálné praxi prosazuje nyní zejména v podnikovém vzdělávání, příp. jako součást vzdělávání vysokoškolského, což může být způsobeno postupným rozšiřováním počítačů od bohatých podniků. Tím, jak se počítače stále více stávají součástí běžných domácností, začínají se objevovat také elektronické kurzy, které jsou určeny pro střední školy. Právě na e-learningové kurzy pro střední školy a jejich specifika se práce soustředí.

Jako součást této práce byl vytvořen jeden takovýto kurz věnovaný tématu algoritmizace a programování. Jeho základním cílem je pokrýt základy oblasti, která je uvedena v dané podkapitole maturitních požadavků ke zkoušce z informačně technologického základu zadávaného MŠMT. Záměrem je vytvořit tento kurz takovou formou, aby budil zájem u jeho uživatelů a stal se pomůckou při výuce pro jejich učitele. Při výuce výpočetní techniky často dochází k situaci, kdy někteří žáci již mají určitou znalost probírané látky, zatímco jiní s ní nemají žádnou zkušenost. Kurz se proto snaží zejména „srovnat“ vstupní předpoklady žáků.

2 Literární rešerše

„Different terminologies have been used for online learning, a fact that makes it difficult to develop a generic definition. Terms that are commonly used include e-learning, Internet learning, distributed learning, networked learning, tele-learning, virtual learning, computer-assisted learning, Web-based learning, and distance learning. All of these terms imply that the learner is at a distance from the tutor or instructor, that the learner uses some form of technology (usually a computer) to access the learning materials, that the learner uses technology to interact with the tutor or instructor and other learners, and that some form of support is provided to learners."[1]

*(*Athabasca University, 2005*)*

E-learning je poměrně rozsáhlou oblastí vzdělávání, která nemá zcela jasné ohraničení. Neshoda panuje dokonce i v samotném zápisu slova – lze se tak setkat s označením e-learning, elearning, E-learning, eLearning či e-Learning. Navíc různí autoři používají různé pojmy s obdobným významem, což dobře ukazuje výše uvedená citace. Tato kapitola proto upřesňuje chápání pojmů užívaných dále v práci a pokouší se stručně shrnout význam e-learningu v edukačních procesech.

2.1 Kořeny e-learningu

Nejčastěji je e-learning zmiňován v souvislosti s distančním studiem, kde nachází široké uplatnění, a proto je tato podkapitola zaměřena na tuto podobu studia. **Distanční vzdělávání** vymezuje slovník (Průcha, Walterová, Mareš, 2003) jako *formu studia zprostředkovanou médii (telefon, rozhlas,*

[1] Volný překlad: *Rozdílná terminologie využívaná pro online vzdělávání má za následek obtížné nalezení jednotné definice pojmu. Běžně se využívá termínů jako e-learning, Internet learning,distributed learning, networked learning, tele-learning, virtual learning, computer-assisted learning, Web-based learning, či distance learning. Všechny tyto pojmy naznačují, že studentům je poskytnuta určitá forma podpory, ale učitel (tutor, instruktor) je vzdálen od studenta, pro distribuci studijních materiálů se využívá technologie (nejčastěji počítač) a technologie umožňuje i vzájemnou interakci mezi studentem a učitelem i studenty na vzájem.*

počítač, zvl. internet, elektronická pošta aj.), která je založena na samostatném studiu účastníků, řízeném specializovanou institucí, bez prezenčního kontaktu studujících s vyučujícími. Některé definice (Jůzová, Kočí a kol., 2006) navíc zdůrazňují předpoklad samostudia a předpřipravených programů.

Za kořeny distančního vzdělávání jsou označovány (Fowler, 2004; PLDG, 2003) korespondenční kurzy těsnopisu, jejichž autorem byl vynálezce těsnopisu Isaac Pitman (1813 – 1897). Jeho studenti opisovali těsnopisem části Bible a své výtvory mu posílali ke kontrole prostřednictvím pošty.

Později bylo možné pro výuku využívat nové komunikační technologie – především televizní a rádiové vysílání (od 20., resp. 40. let minulého století). Ač se může tato forma komunikace zdát na první pohled vzdálená, jedná se o možnost běžně používanou – příkladem lze uvést recepty na jídla popisované v rádiu či televizní kurzy cizích jazyků. Využití telefonu bylo překvapivě mnohem pozdější, a to nejprve pro videokonference (Microsoft, 2005), poté pro připojení počítačů k sítím. Z dnešního hlediska může být úsměvné, ale i poučné, mínění Thomase Alvy Edisona (1847 – 1931), který se domníval (Brdlička, 1995 – 2001), že kinematografie revolučním způsobem změní vzdělávací systém a do značné míry, ne-li zcela, nahradí učebnice.

S nástupem počítačů se o nich začalo uvažovat jako o možných nástupcích učitelů i učebnic – B. F. Skinner (1904 – 1990) vytvořil takzvané programované vyučování (programmed instruction), které spočívalo v rozčlenění látky na malé části, které si student nejprve dokonale procvičil, než mohl pokračovat. Tato metoda zkrachovala, když výzkumy ukázaly nižší efektivitu učení (Dobrovská, 2004). Také její nástupce, tzv. počítačem podporované učení (computer assisted learning, CAL), se ve své původní podobě setkal s rozpaky, když vědomosti žáků získané kombinací klasické výuky s počítači sice nebyly špatné, avšak objevil se silně aversní postoj vůči takto probírané látce (Dobrovská, 2004). Naopak počítače se osvědčují pro multimediální učebnice a encyklopedie, stejně jako pro vyhledávání informací prostřednictvím internetu. Někteří autoři (Barešová, 2003) přisuzují neúspěchy v počátcích zcela lineárnímu uspořádání kurzu, zatímco nynější hypertext umožňuje lepší propojení souvisejících úseků textu.

Nové technologie daly vyniknout tzv. multimediální vyučovací metodě, při níž se využívá různá didaktická technika (počítače, výukové filmy,

magnetofon, tištěné materiály aj.).

2.2 Různé pohledy na e-learning

Již v úvodu bylo uvedeno, že e-learning se překládá jako elektronické učení/vzdělávání. V nejširším významu tak lze konstatovat, že se pojmy e-learning a multimediální vyučovací metoda překrývají.

Proto se lze někdy setkat s rozdělením e-learningu do tří kategorií. Nejširší pojetí klade důraz spíše na „e“, jedná se o vzdělávání využívající „elektronické“ přístroje, obvykle označované jako **TBT** (Technology based training). Většina lidí si však pod e-learningem pravděpodobně představí počítač a určitý kurz na něm, což zdůrazňuje druhá kategorie **CBT** (Computer based training). Nemusí to být však pouze počítač, který zprostředkovává přístup do kurzu – roste obliba malých přístrojů (různé obdoby výkonných mobilních telefonů), které mohou posloužit stejně dobře jako počítače – a proto moderní pojetí si všímá šíření kurzů v prostředí počítačových sítí – **WBT** (Web based training). (Protože mobilní zařízení tvoří velmi přitažlivou výzvu pro e-learning, někdy se lze setkat také s pojmem m-learning. Pod tímto pojmem se však obvykle rozumí spíše teoretická schopnost s e-learningovým systémem pracovat prostřednictvím mobilních zařízení, tedy jakási kompatibilita a rozšíření stávajících systémů.)

Jednotliví autoři se tak ve svých definicích e-learningu snaží co nejvýstižněji pojmenovat „svůj“ pohled na e-learning. Pravděpodobně první kniha napsaná a vytištěná v češtině na toto téma navrhuje definici: „*E-learning je vzdělávací proces, využívající informační a komunikační technologie“* (Barešová, 2003). Pedagogický slovník (Průcha, Walterová, Mareš, 2003) vymezuje e-learning jako *různé druhy učení podporovaného počítačem, zpravidla s využitím moderních technologických prostředků, především CD-ROM*. Dále uvádí, že *elektronické učení se rozšiřuje zejména ve sféře distančního vzdělávání a podnikového vzdělávání*.

Na internetových stránkách českých e-learningových společností se lze setkat s podobnými definicemi. Společnost Kontis výstižně uvádí: *„Řečeno stručně a velmi jednoduše, e-Learning není nic jiného než efektivní využívání Informačních Technologií v procesu vzdělávání. … Tím není vyloučeno, že pro konkrétní situaci může být nejefektivnější IT ve vzdělávání nevyužívat vůbec.“* Na stránkách produktu eDoceo je snaha o komplexnější definici: „*E-learning*

v širším slova smyslu znamená proces, který popisuje a řeší tvorbu, distribuci, řízení výuky a zpětnou vazbu na základě počítačových kurzů, kterým stále častěji říkáme e-learningové kurzy.“

Poslední uvedená definice si všímá faktu, že všechny moderní e-learningová řešení musí umožnit „něco více“ než jen pouhé vytvoření a distribuci nějakého kurzu. Proto jsou často počítačové programy pro e-learning rozděleny do dvou částí. První, která umožňuje zejména vytvořit autorovi kurz, se označuje zkratkou **LCMS** (Learning Content Management System). Druhou je pak **LMS** (Learning Management System), který vytvořené kurzy distribuuje a umožňuje také například sledovat aktivitu studentů.

LCMS **systém pro tvorbu a sestavování obsahu**	**LMS** **systém pro řízení výuky**
▪ týmový proces tvorby obsahu; ▪ správu a znovu používání zdrojů obsahu, ▪ dekompozici a kompozici obsahu na učební jednotky; ▪ dodávání individuálně uzpůsobitelných učebních jednotek; ▪ detailní sledování aktivit uživatelů nad učebními jednotkami; ▪ podporu integrace výukových strategií e-learningu.	• řízení a evidenci všech typů výuky od elektronických asynchronních kurzů, přes virtuální učebny, až po klasickou výuku v učebnách; • centrální katalog všech vzdělávacích akcí (elektronické kurzy, virtuální třídy/videokonference, učebny, externí výuka), registrační procesy, správu zdrojů a financí s tím spojenou; • modelování organizace a kompetencí, evidování dosažených individuálních dovedností; • zpřístupňování vzdělávacích akcí, sledování aktivit jednotlivých uživatelů od souhrnů po detaily, reportování (zaznamenávání) všech typů výukových aktivit společně i jednotlivě; • bohatou sadu synchronních a asynchronních komunikačních kanálů mezi studenty, lektory a manažery vzdělávání, prostředky pro zachytávání, výměnu a sdílení informací a znalostí.

Tab. 1: Rozdíly mezi LMS a LCMS – co musí umožňovat (Pejša, 2004)

Protože tedy existují dva systémy, vyvstává nutnost existence možnosti převádět obsah kurzu mezi více systémy, které mohou být teoreticky od různých výrobců. Jsou proto vytvořeny normy, které by měly e-learningové systémy splňovat, aby bylo teoreticky možné převést kurz (či jeho část) vytvořený v určitém LCMS do libovolného LMS. Respektovaným standardem

je **SCORM** (Sharable Content Object Reference Model = referenční model sdílitelných objektů obsahu, vzniklý díky iniciativě ADL {Advanced Distributed Learning Initiative – Pokročilé distribuované vzdělávání}), který popisuje jednotlivé části e-learningu a klade si za cíl zrychlení vývoje a umožnění znovupoužitelnosti jednotlivých částí obsahu. Další možností je **AICC** (The Aviation Industry CBT {Computer-Based Training} Committee – Komise pro CBT v leteckém průmyslu), který především standardizuje komunikaci s částí vytvořenou v jiném systému.

Samotný e-learning často nepřináší požadované výsledky, a proto se mnohdy využívá začleněn do upravených stávajících edukačních procesů. Stále častěji je tak zmiňován **blended[2] learning** označující výuku, při které dochází ke kombinování *více metod dodávání výuky pro dosažení cílového efektu* (Pejša, 2003).

2.3 Specifika využití e-learningových kurzů na středních školách

I když prakticky každá práce věnovaná e-learningu obsahuje rozepsané jeho výhody a nevýhody, jsou uvedeny i zde. Jedná se však spíše o stručné shrnutí – podrobnější rozepsání je možné najít v množství jiné literatury (např. Net-university, 2006; Barešová, 2003). Důraz je zde kladen na specifika českého středního školství.

2.3.1 Výhody

Mezi základní výhody e-learningu se obvykle zahrnují nižší náklady na vzdělání, široká přístupnost studia či snadná aktualizace studijních materiálů.

Všechny uvedené výhody se však v rámci středních škol příliš neprojeví – do úspory nákladů se započítává například úspora času zaměstnance (pro interní firemní kurzy) nebo výhody úspor z rozsahu. Na středoškolské úrovni není ve většině předmětů ani nutné měnit často studijní materiály a přístupnost studia není příliš velký problém.

Přesto však má e-learning co nabídnout. Jedná se například o efektivní využití času, protože student si může snadno rozvrhnout, kdy bude kurz studovat. Stejně tak si student volí tempo svého studia. To může být výhodné

[2] Blended je možné přeložit jako smíšený či smíchaný.

zejména při příchodu žáků ze základních škol, kde mohl být kladen různý důraz na některé oblasti vzdělávání a vstupní úroveň žáků je tak různá.

Do určité míry e-learning může snížit některá omezení a předsudky a integrovat tak studenty s určitými handicapy (určitý stupeň anonymity umožňuje rovnocenné zapojení například tělesně postižených či národnostních menšin). Navíc se uvádí (Net-university, 2006), že zrak je z hlediska lidského vnímání důležitější než sluch (což je jeden z výrazných rozdílů mezi klasickou výukou a elektronickými kurzy – elektronické kurzy působí zejména na zrak, zatímco klasická výuka spíše na sluch).

2.3.2 Nevýhody

Při využití e-learningu je důležité zvážit vhodnost v konkrétním případě. Pokud by se dokonce eKurzy používaly pouze přímo ve škole, výrazně by poklesly některé uvedené výhody.

Základním předpokladem využití je (v případě využití internetových kurzů) zařízení s dostatečně kvalitním připojením k internetu. Údaje z loňského roku uvádějí (Ambrož, J., 2006), že počítač vlastní 36 % domácností, připojení k internetu však jen 27 % (navíc z tohoto čísla se nepříliš vhodné vytáčené připojení drží nad třiceti procenty). Lepší zprávou je, že připojeny jsou zejména domácnosti se studenty a existuje předpoklad dalšího růstu připojených domácností (průměr EU15 (jinou metodikou) činil 53 % oproti 29 % v ČR).

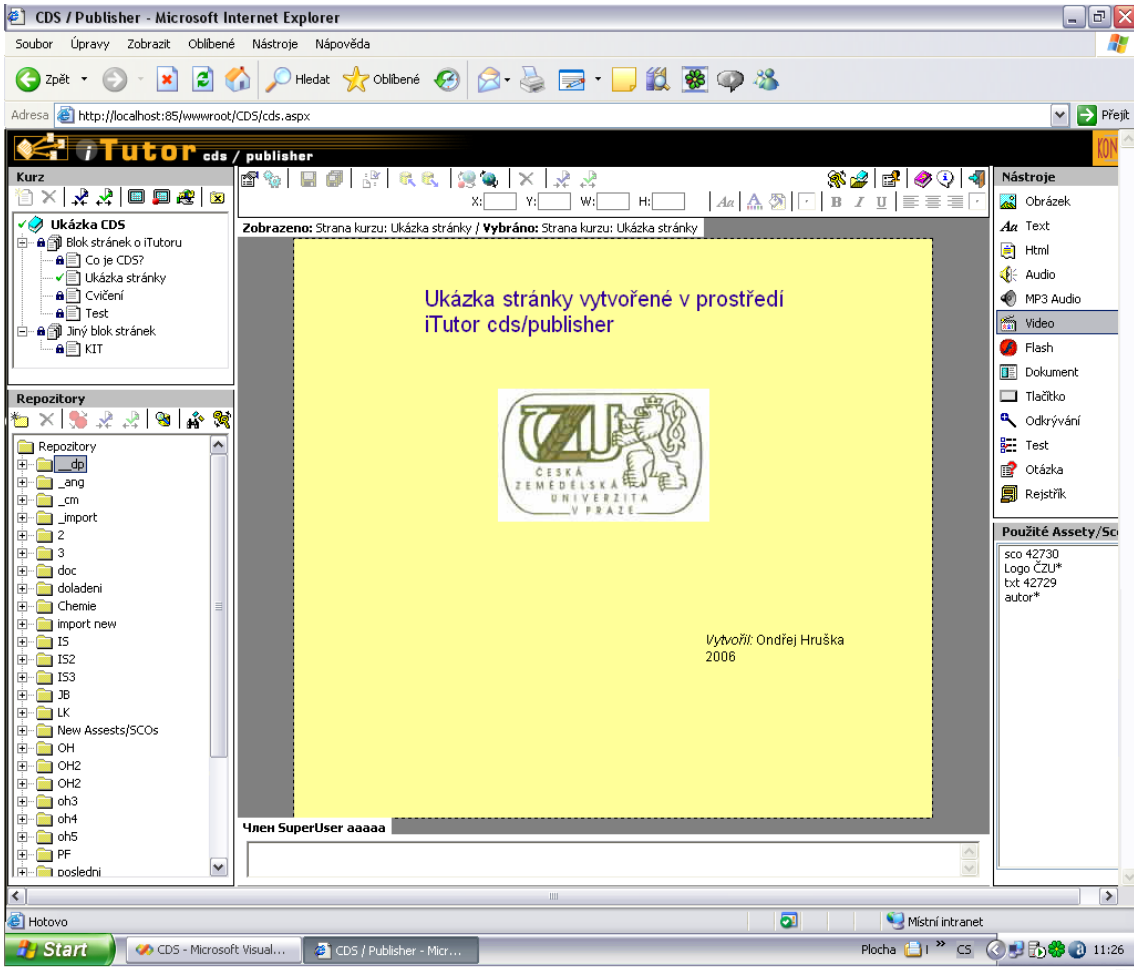
Navíc se elektronický kurz nemusí hodit pro všechny studenty, předměty či pedagogické směry. Ne každý student je schopen studovat samostudiem a přes internet probíhá přece jenom odlišný způsob například socializace. Rovněž edukační proces ohrožuje zneužití anonymity i fyziologicko-psychologické hranice práce s počítačem. Nepříjemnosti může rovněž způsobit nekompatibilita mezi různými e-learningovými systémy, protože není možné vždy přenést již vytvořený kurz do jiného e-learningového systému.

2.4 Software pro e-learning

V následující kapitole budou podrobně rozvedeny prostředky

(počítačové programy), které byly využity při tvorbě kurzu. Ačkoli kurz, který je praktickou částí práce, nebyl zpracován pomocí softwaru, který by byl určen primárně pro e-learning, existuje mnoho speciálně vytvořených programů (e-learningových systémů). Sice je možné se bez těchto programů obejít, je to však cesta pracná, kdy se tvůrce kurzu musí poměrně dobře vyznat zejména v technologiích pro tvorbu webu. Již v kapitole 2.2 bylo uvedeno, že software pro e-learning lze rozdělit do dvou velkých skupin: LCMS (pro tvorbu obsahu) a LMS (pro řízení výuky).

Jako velmi jednoduchý LCMS si lze představit například PowerPoint, který umožňuje uživatelům jednoduše vytvářet a formátovat texty či obrázky. Existuje dokonce program Macromedia Breeze Presenter, který je jakýmsi „doplňkem“ PowerPointu a umožní převést existující prezentaci a případně ji doplnit o zvuky či animace. (Více například na http://el.lf1.cuni.cz/podporabreeze/.) Další možnost představuje iTutor CDS, který umožňuje například správu verzí jednotlivých objektů (obrázků, zvuků, textů apod.), paralelní práci na obsahu kurzu (uzamykání objektů), či import dat z PowerPointových souborů. Poněkud odlišný přístup představuje Autor od společnosti eDoceo, který je na rozdíl od předchozích programů přístupný zdarma, avšak nabízí jen vytvoření struktury kurzu z existujících materiálů. (Struktura kurzu však byla vytvořena i tímto nástrojem pro případný export do systému LMS eDoceo.)



Obrázek 1: Editor iTutor CDS/Publisher.

LMS se zabývá správou již existujících kurzů, ale také vlastní prací se studenty a správou celého edukačního procesu. Z hlediska elektronických kurzů se jedná například o sledování práce jednotlivých uživatelů (statistiky jejich úspěšnosti, doby studia aj.), ale také třeba umožněním vzájemné komunikace studentů i učitelů. Obvyklá je u těchto systémů modulární architektura, která umožňuje nainstalovat jen určitou část používaných funkčností. Do určité míry je možné suplovat LMS pomocí více různých programů (např. ICQ nebo Skype pro komunikaci), avšak uživatelé pak musí být schopni ovládat množství různých prostředí.

2.5 Proces tvorby e-learningového kurzu

Při vytváření elektronického kurzu by se nemělo jednat o pouhý převod textových dokumentů do elektronicky přístupné podoby. Je třeba, aby kurz své uživatele motivoval k jeho studiu a snažil se ho zapojit do edukačního procesu. (Studenti by v žádném případě neměli být jen „pasivními konzumenty moudrostí“ – ve své podstatě se jedná o aplikaci aktivizujících metod v elektronické oblasti). Kvalitní elektronický obsah **(e-content)** může autor vytvářet sám, nebo se může využít nabídek na zpracování obsahu (tyto služby poskytuje prakticky každá společnost zabývající se e-learningem).

Představu o tom, co vše je při vytváření kurzu potřeba, je možné získat již z krátkého představení rolí obvyklých (Zlámalová, 2002) při vytváření elektronického kurzu.

Pochopitelně zde existuje autor, který vytváří obsah. Ten spolupracuje s pedagogem, který dohlíží na didaktickou stránku kurzu. Velké projekty zahrnují také manažera (starajícího se o řízení projektu) a garanta odpovědného za kvalitu.

Při přepisu textů lze využít programátora a grafika (případně se může jednat o další osoby odpovědné například za zvukovou stránku kurzu nebo obsažené animace). Po vytvoření kurzu by mělo následovat otestování (nejprve testery na „softwarovou funkčnost“) a poté případné úpravy s ohledem na oponenta i prvotní čtenáře (studenty).

Chod celého kurzu by měl zajišťovat správce, velmi důležitá je i role tutora (*Tutor je metodický zprostředkovatel distančního studia a hodnotitel průběžných výsledků.* (Zlámalová, 2002)). Protože tutor by měl být zejména „rádce a průvodce“ kurzem „s plnou důvěrou studentů pro případné dotazy“, vlastní hodnocení kurzu by měla obstarávat jiná osoba – examinátor.

3 Vymezení a zpracování tématu

3.1 Význam výuky algoritmizace a programování

Bohužel v současné době musí výuka algoritmizace a programování stále ještě často „bojovat o své místo na slunci“. Je však třeba si uvědomit, že její význam na středoškolské úrovni nespočívá ve výchově programátorů, ale v pochopení toho, jak počítačové programy fungují. Navíc algoritmizace úlohy (tedy vlastně rozdělení řešení problému do jasné posloupnosti kroků) může nalézt uplatnění i v „nepočítačových“ oblastech. Programování by tedy mělo být zaměřeno na rozvoj myšlení žáků, spíše než na konkrétní znalosti například z oblasti syntaxe a sémantiky konkrétního programovacího jazyka. Nejprve je třeba začít teorií, která může být pro někoho „nezáživná“, ale věřím, že vlastní programování je tvůrčí činnost, která stojí na obecně platných principech. A tvůrčí práce je vlastně hra s počítačem, jež snad ani nemůže být nezábavná.

Velkým přínosem pro tuto oblast tak mohou být otázky z Katalogu požadavků k maturitní zkoušce (MŠMT, 2006), kde oblast „Algoritmizace a základy programování, makra“ tvoří jeden z osmi tematických okruhů. Je však otázkou, zda bude tento okruh skutečně jednou osminou v otázkách zastoupen (již v katalogu je zastoupení drobně sníženo na 10 %) a zda mu bude věnována také osmina hodinové dotace předmětu.

Tematické okruhy	%
Informace a informační zdroje	10
Hardware a software, sítě	10
Využití internetu	15
Textový editor	15
Tabulkový kalkulátor	15
Počítačová grafika, prezentace, tvorba webových stránek, multimédia	20
Databázové aplikace	5
Algoritmizace a základy programování, makra	10

Tab. 2: Procentuální zastoupení tematických okruhů u maturitní zkoušky z informačně technologického základu (MŠMT, 2006)

3.2 Volba programovacího jazyka pro příklady

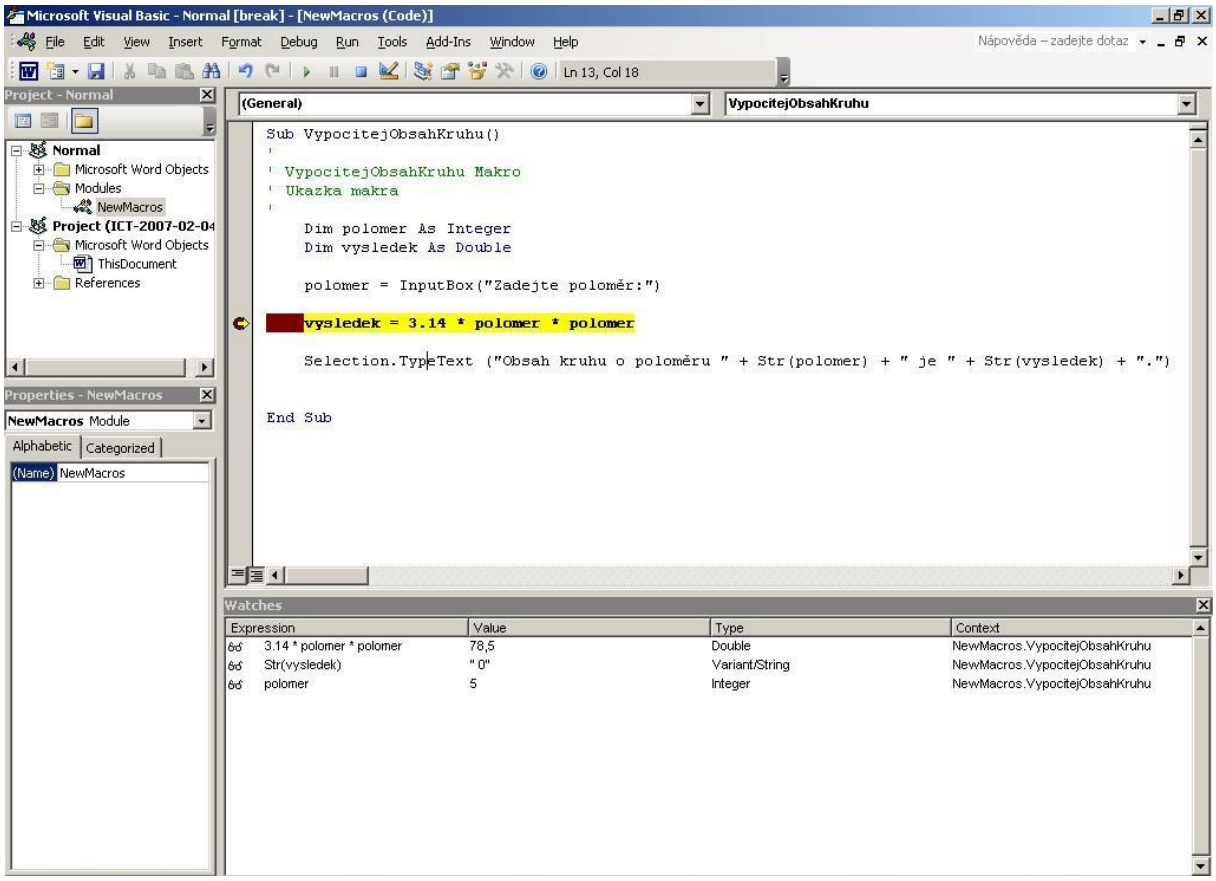
Nejednoduchou otázku při výuce programování tvoří volba programovacího jazyka. Velkým ohrožením pro vlastní programování je pochopitelná snaha učitelů učit takový jazyk, který sami dobře znají. Bohužel programovací jazyky jako například C, Basic nebo Pascal jsou ve své původní podobě prakticky „mrtvé“. (Podobně třeba latina je jazyk, který je pro určité oblasti dobré znát, ale není třeba, aby ho všichni uměli. A pokud by výuka cizích jazyků byla zaměřena na latinu, těžko by nacházela své zájemce.) Nelze předpokládat, že studenty zaujmou složitě vytvářené textové programy, které v porovnání s programy Windows vypadají opravdu nepřitažlivě. Sám jsem se jako student (i na vysoké škole) setkal s tvrzením, že jde především o principy a pochopení programování, které je pak podobné. Toto tvrzení je sice pravdivé, ale studenti tráví zbytečně mnoho času zapisováním příkazů a vzhled výsledného programu často neodpovídá vynaložené snaze.

Přitom současné programovací jazyky jsou z těch původních často odvozené a mají základní programové konstrukce zcela shodné. Navíc pro ně existují vývojová prostředí zdarma (například Delphi Personal Edition nebo Express Edition pro .NET), která obsahují nástroje snižující množství času, které by bylo nutné věnovat například obyčejné změně barvy a fontu písma.

Kurz byl vytvářen především s ohledem na požadavky „státních maturit“ i s respektováním pravděpodobného softwarového vybavení většiny škol. Proto bylo upuštěno od vytváření samostatných programů a na místo toho bylo zvoleno prostředí textových editorů. Současné textové editory totiž většinou obsahují možnost vytvářet si v jejich rámci nové funkčnosti (tzv. makra).

Navíc mají poměrně jednoduše ovladatelné prostředí pro tvorbu těchto maker a knihovnu mnoha užitečných funkčností. Rovněž je možné se v tomto prostředí setkat s moderními prvky programování jako je objektový přístup nebo vizuální tvorba programu. Jazyk, který využívá v České republice nejrozšířenější MS Office, vychází z jazyka Basic (přesnější označení je Visual Basic for Application – zkratka VBA). Obdobný jazyk i ovládání lze nalézt i ve volně šiřitelném OpenOffice.org. Závěrečná kapitola kurzu proto seznamuje se základy jazyka Basic a ukazuje jejich využití v rámci maker Wordu. Navíc oblast maker v kancelářských aplikacích je oblast, se kterou se

studenti velmi pravděpodobně i později setkají.



Obrázek 2: Editor jazyka Visual Basic v MS Office, ukázka krokování programu

3.3 Nástroje využité při tvorbě kurzu

Kurz vypracovaný jako součást práce byl vytvořen jako propojené internetové stránky. Pro vývoj internetových stránek existuje v dnešní době celá řada programů. V této kapitole je pojednáno o těch, které byly využity – snahou při vytváření celého kurzu bylo používat takové programy, které jsou k dispozici zdarma (přesné a aktuální informace o licenci produktů je možné najít v jejich specifikaci na internetu).

Volba kurzu jako HTML[3] stránek má velkou výhodu v poměrně snadné dostupnosti uživatelům v prostředí internetu. Pravděpodobně každý, kdo má přístup k internetu, má i prohlížeč, který by umožňoval přístup do kurzu. Na druhou stranu mohou nastat určité problémy způsobené nekompatibilitou

[3] HTML je zkratka z anglického HyperText Markup Language (značkovací jazyk pro hypertext). Primárně je určen pro popis struktury dokumentu, což umožňuje uzavíráním úseků dokumentu do značek, čímž určuje jeho části (nadpisy, odstavce apod.). Využívá se pro vytváření internetových stránek.

prohlížečů. Kurz byl proto odzkoušen pro prohlížeče MS Explorer (MSIE 6.0.x) a Mozzila FireFox 1.5.x, které pokrývají většinu trhu. Zároveň byly v co nejvyšší míře respektovány standardy W3C[4], což by mělo zaručit funkčnost stránek i v jiných prohlížečích.

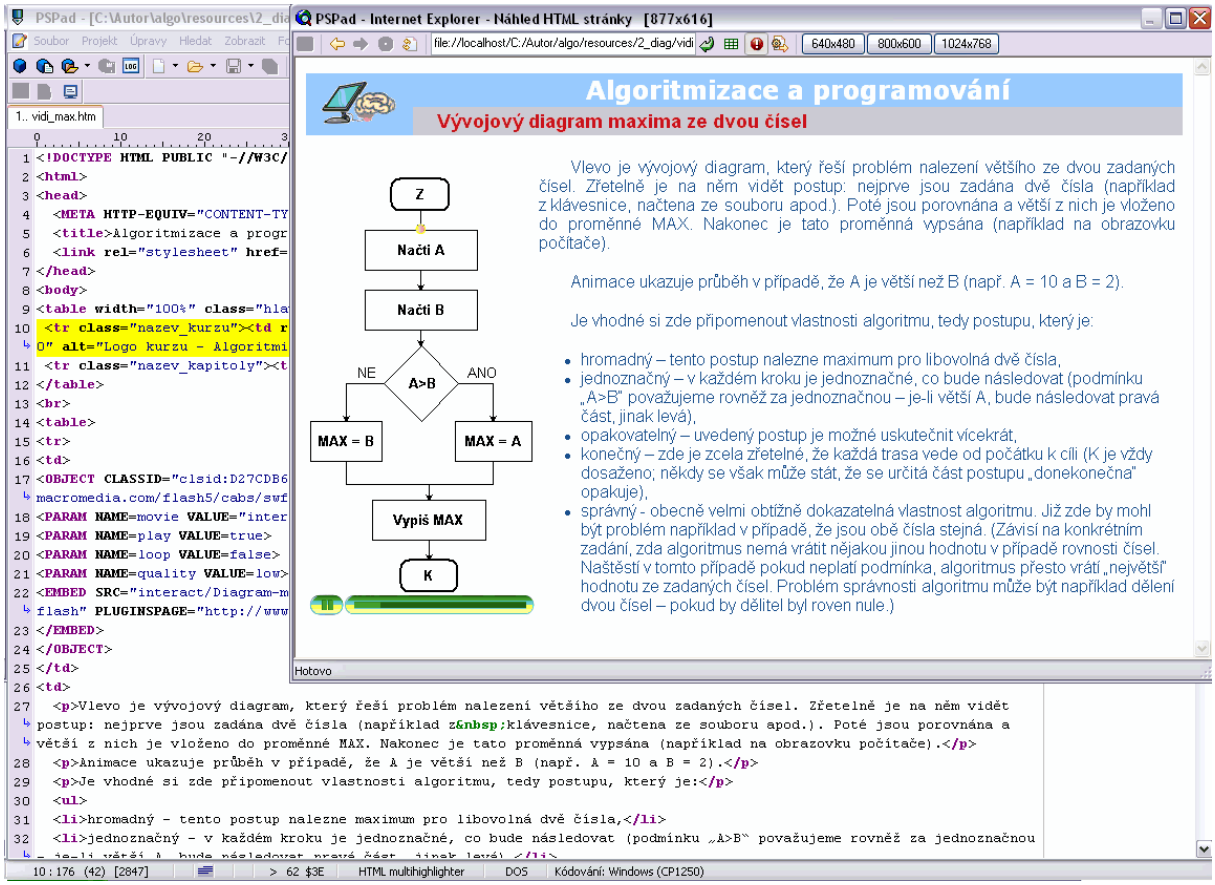
3.3.1 Textový editor

Kurz se, jak již bylo uvedeno, skládá z HTML stránek. Ty je možné vytvářet například v textových editorech v módu WYSIWYG[5], kdy pak ale výsledné stránky obsahují množství „redundantního" textu a nejsou často příliš kompatibilní s různými prohlížeči. Jednotlivé stránky byly proto napsány přímo v HTML, což odstraňuje zmíněné problémy, ale na druhou stranu vyžaduje dobré znalosti problematiky tvorby webových stránek.

Pro vytváření kurzu byl proto využit český produkt PSPad Editor. Součástí kurzu byl rovněž dotazník vytvořený v prostředí ASP.NET (produktem Web Developer).

[4] The World Wide Web Consortium (W3C) je konsorcium, které se stará o webové standardy (týká se technologií, jako jsou značkovací jazyky HTML, XML nebo kaskádové styly CSS).

[5] WYSIWYG je akronym anglické věty „What you see is what you get", tedy česky „co vidíš, to dostaneš". Tento způsob je využíván například v textových editorech a označuje především skutečnost, že vytištěný text bude stejný jako text zobrazený na monitoru. (V tomto konkrétním případě se „na pozadí generují" HTML značky běžným formátováním textu.)



Obrázek 3: Stránka a její HTML kód.

3.3.2 Zpracování obrázků

Pro oblast vytváření a úpravy obrázků existuje velmi široká nabídka zdarma dostupných programů. Asi nejjednodušší možnost představuje program Malování, který je součástí operačního systému Windows. Pokročilejší malování a úpravy umožňuje GIMP, který navíc není vázán pouze na platformu Windows. Střední cestu představuje 602Photo nebo Microsoft Office Picture Manager (je součástí MS Office).

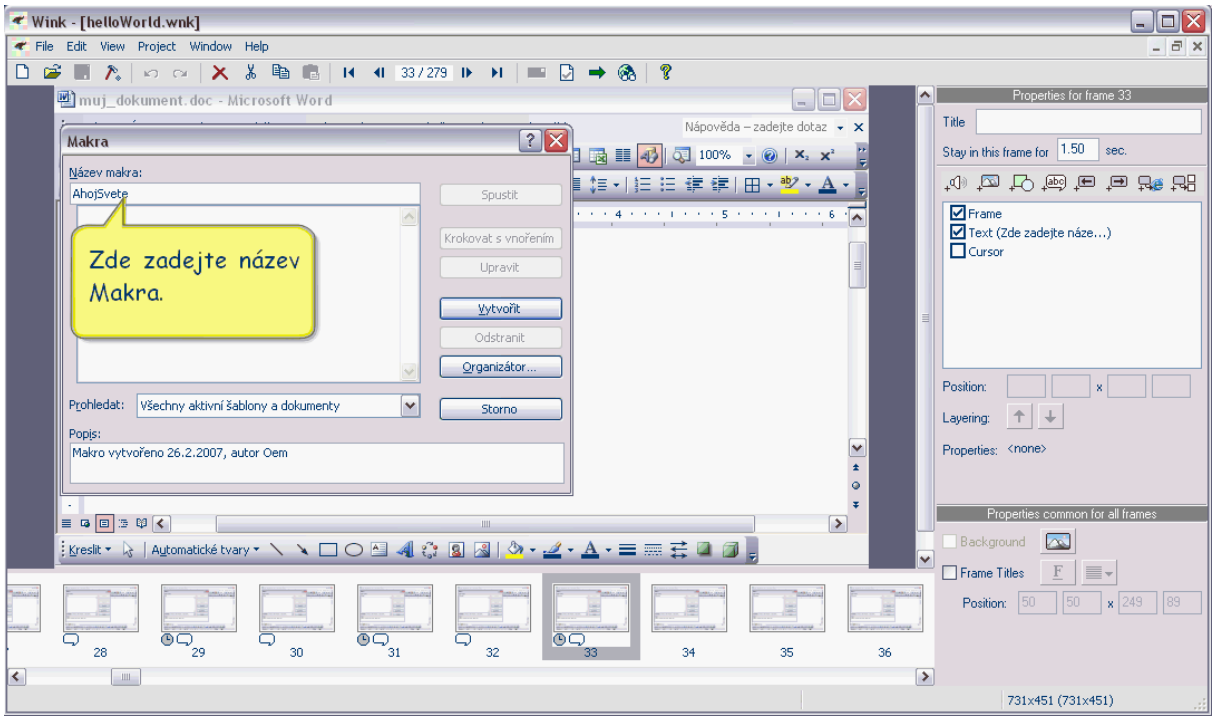
Kromě výše uvedeného softwaru byl využit také Diagram Designer, který umožnil rychlé nakreslení vývojových diagramů ve třetí kapitole kurzu.

3.3.3 Vytvoření animací a úpravy zvuků

Pro vytvoření skutečně interaktivní animace v prostředí internetu neexistuje bohužel mnoho nástrojů ani formátů. Některé části kurzu obsahují programy v JavaScriptu[6]. Je v něm vytvořeno například skrývání a odkrývání částí stránek kurzu i některé testy.

Skutečně interaktivní animace lze vytvářet pomocí technologie Flash. Jedná se o velmi mocný nástroj, jehož vývojová prostředí jsou bohužel prakticky vždy placená (byla v něm vytvořena animace na straně 2 – „Úvodní slovo“). Přesto už existují zdarma přístupné programy, které umožňují export do tohoto formátu. Animace (ve druhé a třetí kapitole kurzu) byly vytvořeny ve skutečně povedeném programu Wink.

Program Wink umožňuje i určitou práci se zvukem, ale prakticky profesionální úpravy umožňuje Audiocity, jehož služeb bylo využito pro nahrání a úpravu zvukových souborů MP3[7] v kurzu.



Obrázek 4: Úprava jednotlivých snímků animace v programu Wink.

[6] JavaScript (včetně jeho „variant“ JScript a ECMAScript) je nejrozšířenější programovací jazyk pro vytváření „aktivního chování“ na straně klienta (prohlížeče). Místo označení „program“ se někdy užívá slovo „skript“.

[7] MP3 je formát ztrátové komprese zvukových souborů, který snižuje velikost těchto souborů odstraněním zvuků, které jsou pro člověka „zbytečné“, protože jsou jen obtížně postřehnutelné.

3.4 Obsah elektronického kurzu

Jak již bylo uvedeno, byl kurz vytvářen především s ohledem na požadavky MŠMT na maturanty. Učební látka byla rozdělena do tří kapitol tak, aby bylo možné probrat jednu kapitolu během běžné vyučovací hodiny.

Po části, která se snaží studenty motivovat ke studiu kurzu i je seznamuje s obsahem kurzu, následuje první kapitola věnovaná základní terminologii oblasti. Studenti se zde dozvědí, co je to algoritmus, co program, k čemu slouží programovací jazyk i jak probíhá vývoj programů. Po absolvování druhé kapitoly budou studující znát princip vývojových diagramů (tato oblast sice není součástí okruhů od MŠMT, ale je důležitá pro pochopení zápisu algoritmu v programovacím jazyce) a získají představu o objektovém přístupu. Poslední kapitola už není tolik teoretická a zaměřuje se na znalosti, dovednosti a návyky potřebné při tvorbě programů. Po jejím nastudování budou studenti schopni vytvářet jednoduchá makra, a to jak prostřednictvím jejich záznamu v kancelářských balících, tak zápisem zdrojových kódů.

Bylo by pochopitelně chybou, kdyby se kurz skládal pouze z textů a obrázků, či animací a zvuků. Obsahuje proto množství interaktivních prvků, které umožňují "vtažení uživatelů do kurzu". Někde se například studenti mají zamyslet nad určitým problémem a jeho řešení se jim zobrazí až po kliknutí na odkaz.

Každá kapitola navíc končí stránkou, kde si studenti mohou otestovat, jak pochopili požadované vědomosti. Jedná se například o úkol seřadit etapy při vývoji softwaru – studenti mají za úkol pomocí tažení myší seřadit tyto činnosti ve správném pořadí. Po druhé kapitole pak následuje doplňování textu jako odpovědi na otázku (tento způsob má tu nevýhodu, že každý překlep se projeví jako nesprávná odpověď, na druhou stranu ho nelze splnit „pouhým namátkovým klikáním“). Test v poslední kapitole pak obsahuje otázky na všechny části kurzu v asi nejobvyklejší podobě – výběru správné odpovědi z více nabízených variant.

3.5 Názory studentů na kurz

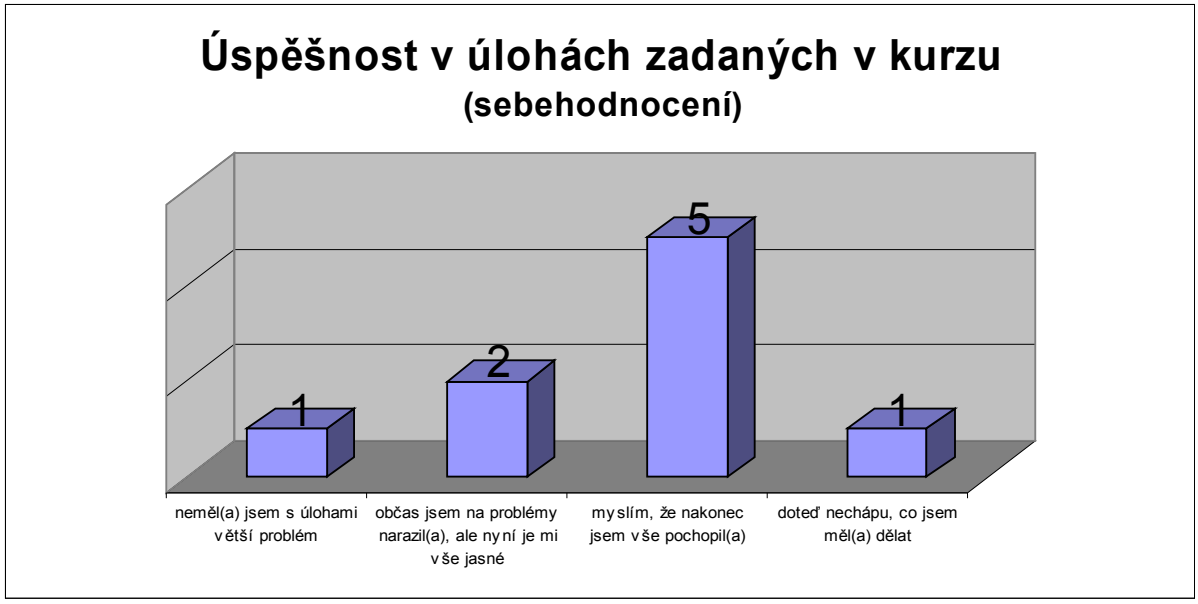
Tato kapitola obsahuje shrnutí výsledků dotazníku, který byl vyplněn

studenty jako zpětná vazba po absolvování kurzu.

Kurz byl vyzkoušen jednou třídou na střední škole ekonomického typu. Po dohodě s tamějším vyučujícím byly první dvě kapitoly kurzu předloženy studentům ke studiu během první dvouhodinové vyučovací jednotky. Ve druhé dvouhodinové vyučovací jednotce pak byla nejprve se studenty zopakována látka z předchozí hodiny a poté studenti sami pokračovali ve studiu kurzu. Na závěr vyplnili krátký dotazník týkající se kurzu (dotazník je zobrazen v příloze práce). Dotazník byl přístupný v elektronické podobě a výsledky tak byly přístupné ihned.

Na otázky odpovědělo celkem 9 žáků čtvrtého ročníku dne 30. března 2007 (jednalo o 6 studentek a 3 studenty). Jak z dotazníku vyplynulo, nikdo z nich neměl předchozí zkušenost s elektronickým vzděláváním.

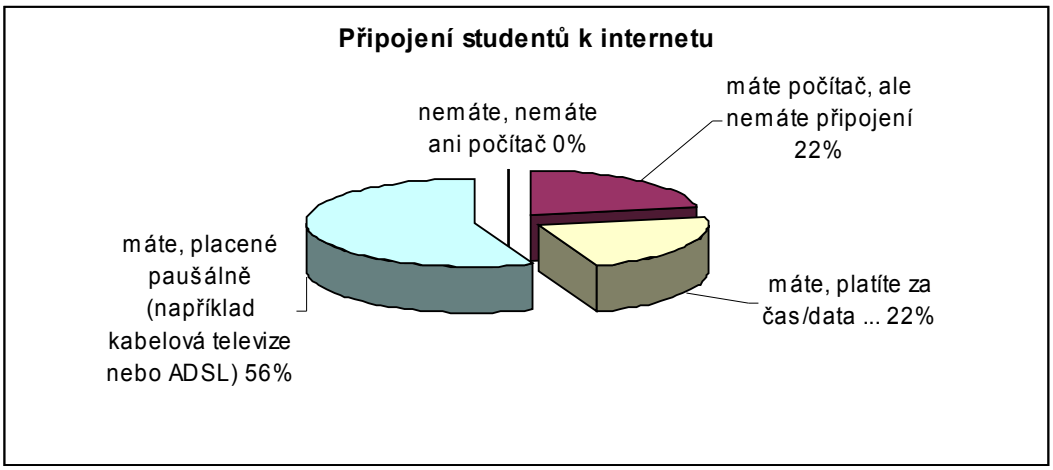
Všichni studenti mají počítač, z čehož pouhé dva počítače nejsou připojeny k internetu. Naopak „nepřetržité“ připojení k internetu měla nadpoloviční většina (5) počítačů. U počítače pak dle odpovědí tráví studenti v průměru 33,11 hodin týdně.



Graf 1: Výsledky dotazníku – sebehodnocení úspěšnosti v úlohách.

Celkově studenti hodnotí kurz jako zajímavou zkušenost (jedinou výjimkou byla odpověď „bylo to skvělé“). S obsahem kurzu (probíranou látkou) studenti občas měli problémy (4 dotazovaní zvolili možnost „některé části byly pro mě nesrozumitelné“; 3 odpověděli „až na pár výjimek

srozumitelné“ a po jednom hlasu dostaly obě krajní odpovědi („rozuměl(a) jsem bez problému“ všemu a „v kurzu jsem se špatně orientoval(a) a často nechápal(a) souvislosti“)). Úspěšnost studentů v kurzu až na dvě výjimky, kopírovala hodnoty z předchozí otázky. Je možné, že v hodnocení kurzu se může projevit také vztah k počítači obecně – žákyně s nejmenším počtem hodin u počítače ohodnotila kurz nejhůře. Celkově si myslím, že hodnocení není špatné, vzhledem k tomu, že základy programování patří v rámci středoškolské informatiky spíše k náročnějším okruhům.



Graf 2: Výsledky dotazníku – otázka na vlastní připojení k internetu.

I přes malý počet respondentů se tyto údaje příliš neliší od výsledků podobného průzkumu, který byl proveden mezi 100 studenty prvních ročníků ČZU v Praze v loňském roce (Hruška, 2006) a nabízí se naopak zajímavé srovnání.

Průměrná doba u počítače dosáhla v případě vysokoškoláků 23 hodin týdně (stojí ovšem za zmínku, že pro prezenční studenty to bylo pouze 11). Tehdy celkem 69 % procent odpovědělo, že má připojení k internetu placené za čas/dat a dalších 16 % pak připojení paušální. Na otázku „Myslíte si, že kvalitně napsaná literatura může nahradit učitele“, která je obdobou šesté otázky, odpovědělo 68 % studentů kladně.

.

4 Závěr

Již v úvodu bylo zmíněno, že mnoho lidí nahlíží na počítač jako na „rychlého idiota“. Může jim proto přijít zvláštní, že by je „idiot“ mohl učit. Přesto existuje e-learning (elektronické učení/vzdělávání), který již přestává být pouhým moderním trendem, ale nachází praktické využití.

Práce se zaměřuje na možnosti uplatnění e-learningu v rámci středoškolského studia, kde jeho využití nyní stále není příliš běžné. Zmiňuje také obecné informace o elektronickém vzdělávání – jeho vývoj, výhody i nevýhody či uvádí příklady e-learningových systémů.

Jako součást práce byl vytvořen elektronický kurz určený pro střední školy. Studenti po jeho absolvování získají základní přehled v oblasti algoritmizace a programování. (Navíc obsah kurzu vychází z Katalogu požadavků k maturitní zkoušce).

V práci je popsáno, jak byl kurz vytvořen, i které programy byly pro jeho tvorbu využity.

Celý kurz byl studován studenty střední školy ekonomického směru, kteří po dokončení studia kurzu vyplnili krátký dotazník týkající se jejich názorů na elektronické vzdělávání obecně i tento kurz. Zpracování těchto dotazníků je součástí práce.

5 Seznam použité literatury

i) Athabasca University, 2004 [cit. 2007-01-20]: *Theory and Practice of Online Learning* [on-line]. Dostupné z: http://cde.athabascau.ca/online_book/

II) PRŮCHA, J., WALTEROVÁ, E., MAREŠ, J., 2003: *Pedagogický slovník*, 4. aktualizované vydání, Praha: Portál, ISBN 80-7178-772-8

iii) JŮZOVÁ, J., KOČÍ A. a kol., 2006: *Universum : encyklopedie pro 21. století*, Praha: Euromedia Group, ISBN-80-242-1755-4

iv) PLDG. *History of distance education* [on-line] 2003, [cit. 2005-09-01]. Dostupné z: http://www.pldg.pl/pldg/portal/media-type/html/user/anon/page/article/node_id/2-25

V) FOWLER, F. 2004: *Historical Overview of Distance Education* [on-line]. [cit. 2005-09-01]. Dostupné z: http://www.people.memphis.edu/~ffowler/overview.html

vi) Microsoft® Encarta® Online Encyclopedia 2005. *Distance Education* [on-line].[cit. 2005-09-01]. Dostupné z: http://ca.encarta.msn.com © 1997-2005 Microsoft Corporation. All Rights Reserved

VII) BRDLIČKA, B., 1995 – 2001: *Role internetu ve vzdělávání: Studijní materiál pro učitele snažící se uplatnit moderní technologie ve výuce* [on-line]. [cit. 2005-09-01]. Dostupné z: http://omicron.felk.cvut.cz/~bobr/hmind/citaty/citaty.htm, ISBN 80-239-0106-0

VIII) DOBROVSKÁ, D., 2004: *Pedagogická a psychologická příprava učitelů odborných předmětů*. 1. vydání, Praha: ISV nakladatelství, ISBN-80-86642-33-X

IX) BAREŠOVÁ, A, 2003: *E-Learning ve vzdělávání dospělých*. 1. vydání. Praha: Nakladatelství VOX, ISBN 80-86324-27-3

X) KONTIS, s. r. o.: *Co je to e-Learning?* [on-line]. Dostupné z: http://www.e-learn.cz/uvod_coje.asp?menu=elearning&submenu=coje

XI) PEJŠA, J., 2004: *LCMS a LMS, vývoj kurzů* [on-line]. [cit. 2007-12-29]. Dostupné z: http://www.e-learn.cz/soubory/LMS_LCMS.pdf

XII) PEJŠA, J., 2003 : *E-learning – trendy, měření efektivity, ROI, případové studie* [on-line]. [cit. 2006-03-05]. Dostupné z: http://www.e-learn.cz/soubory/e-learning_trends_ROI.pdf

XIII) NET-UNIVERSITY, 2006. *Výhody a nevýhody e-learningu* [on-line]. [cit. 2007-04-02]. Dostupné z: http://www.net-university.cz/dtext.php

XIV) AMBROŽ, J., 2006: *Češi a připojení k Internetu, léta páně 2006* [on-line]. [cit. 2007-04-02]. Dostupné z: http://www.lupa.cz/clanky/cesi-a-pripojeni-k-internetu-leta-pane-2006/

XV) ZLÁMALOVÁ, H., 2002: *Principy distanční vzdělávací technologie a možnosti jejího využití v pedagogické praxi na technických vysokých školách* [on-line]. [cit. 2007-04-09]. Dostupné z: http://icosym.cvut.cz/telel/zlamalova.html

XVI) HRUŠKA, O., 2006: *Tvorba e-learningového kurzu pro předmět Informatika II*. Diplomová práce, Provozně ekonomická fakulta ČZU, Praha

xvii) Ministerstvo školství, mládeže a tělovýchovy, 2006: *KATALOG POŽADAVKŮ K MATURITNÍ ZKOUŠCE: INFORMAČNĚ TECHNOLOGICKÝ ZÁKLAD* [on-line]. [cit. 2006-03-05]. Dostupné z: http://www.cermat.cz/novamaturita/katalogy/Katalog_ITZ_10022006.pdf

6 Rejstřík

Přílohy

MATURITNÍ POŽADAVKY (specifické cíle) ke zkoušce z informačně technologického základu zadávané MŠMT

(pouze 8. okruh – více viz. http://www.cermat.cz/)

Maturitní požadavky představují konkrétní požadavky k maturitní zkoušce z informačně technologického základu. Vznikly promítnutím očekávaných znalostí a dovedností do tematických okruhů a jsou podle tematických okruhů členěny. Maturitní požadavky jsou formulovány pomocí aktivního slovesa, které navazuje na úvodní formulaci *„Žák dovede“*. Tato formulace pro lepší přehlednost není před každým požadavkem uváděna.

...

8. Algoritmizace a základy programování, makra

8.1. algoritmizace úlohy, vlastnosti algoritmu

8.1.1. vysvětlit postup vzniku počítačového programu (analýza zadání, návrh řešení, algoritmizace řešení, zápis programu a jeho ladění, podpora a údržba programu)
8.1.2. vysvětlit pojem a účel tzv. myšlenkové mapy a vytvořit myšlenkovou mapu zadané oblasti
8.1.3. vysvětlit pojem algoritmus a jeho vlastnosti (hromadnost, podmíněnost, opakovatelnost, konečnost)
8.1.4. algoritmizovat jednoduchou úlohu

8.2. základní programové a datové struktury

8.2.1. vysvětlit základní příkazy strukturovaného programování (příkaz, posloupnost příkazů [složený příkaz], podmíněný příkaz, cyklus s podmínkou na začátku a na konci, cyklus s pevným počtem opakování a základní programové struktury (procedury a funkce)
8.2.2. vysvětlit pojmy proměnná, identifikátor a datový typ a znát základní typy proměnných (znak, řetězec, celé číslo, reálné číslo, logická hodnota)
8.2.3. rozumět pojmu syntaxe programovacího jazyka
12

8.3. přehled současných způsobů tvorby programů (objektové a vizuální programování)

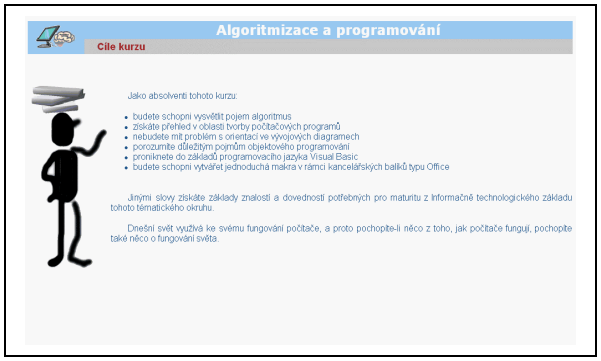
8.3.1. vysvětlit princip objektového programování (zapouzdření proměnných, procedur a metod do objektů, řízení tokem událostí)
8.3.2. vysvětlit princip vizuální tvorby programu (výběr připravených komponent a programování reakcí na události, které jsou s nimi spojené)

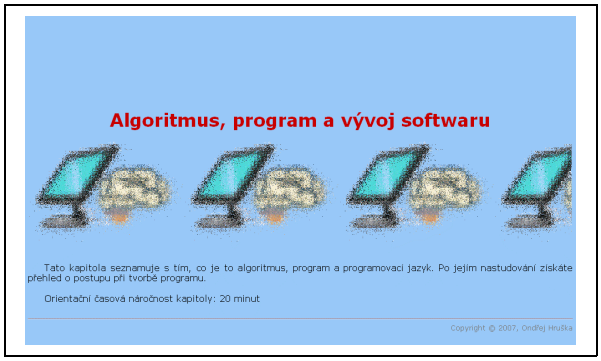
8.4. záznam a spuštění makra

8.4.1. vysvětlit pojem makro
8.4.2. zaznamenat jednoduché makro (např. v textovém editoru)
8.4.3. pojmenovat a spustit dříve zaznamenané makro

Náhled stran kurzu

Kompletní kurz je přístupný prostřednictvím internetu na adrese http://algoritmizace.asp2.cz/. Zde jsou zobrazeny jednotlivé strany kurzu, avšak pochopitelně není takto možné zachytit interaktivní prvky (animace apod.).





Algoritmizace a programování

Dělba práce při vytváření programu

Proces psaní příkazů v programovacím jazyce se nazývá **programování**. Aby programování skutečně vedlo k požadovanému výsledku, podílí se na vytvoření programu obvykle více osob.

V prvé řadě je třeba přesně určit, co bude program umět – o tuto činnost se obvykle stará **analytik**. Analytik poté vypracuje určité zadání (nejprve zjistí a popíše řešený problém a poté naznačí jeho řešení), které předá **vývojáři** (označovanému také jako **programátor**). Vývojář podle popisu vytvoří program (přepíše řešení do programovacího jazyka). Zápis programu v programovacím jazyce se nazývá **zdrojový kód**. Poté je spuštěn program, který dokáže přeložit tento zdrojový kód do jazyka počítače, a vznikne tak spustitelný program. Prvotní otestování programu se nazývá jeho **ladění**.

Vytvořením spustitelného programu práce ani zdaleka nekončí. Tento program dostane **tester**, který mu zadává různá data a zkoumá chování programu. Pokud objeví chybu, pak ji předá zpět programátorovi k dořešení.

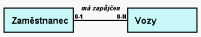
Poté je program předán uživatelům. V průběhu práce s programem může docházet k určitým požadavkům na změny – například v důsledku nakoupení nového počítače nebo pro vytvoření další funkce, kterou původní program neobsahoval. Stejně tak mohou uživatelé chtít, aby jim bylo vysvětleno, jak mají software používat. Této části „života programu" se říká **údržba a podpora**.

Algoritmizace a programování

Myšlenková mapa

Protože problém řešený programem může být velice složitý, využívá se často k jeho popisu kombinace slov a obrázků. Tuto kombinaci lze nazvat určitým typem **myšlenkové mapy** – ta se snaží zachytit všechny důležité předměty i vztahy mezi nimi graficky i verbálně.

Pojem myšlenková mapa se v programátorské praxi příliš nepoužívá, používají se spíše různá označení diagramů (například DFD, E-R Diagram, nebo Use Case).



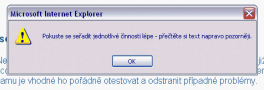
Například tento jednoduchý obrázek ukazuje diagram, který se využívá při ukládání dat do počítače (přesněji řečeno se jedná určitou variantu E-R diagramu používaného v databázích). Vyjadřuje skutečnost, že zaměstnanec může mít zapůjčeno více vozů (nebo také žádný), a že každý vůz může (nebo nemusí) být zapůjčen zaměstnancem.

Místo složité věty se tak používá jednoduchý obrázek – tento přístup je blízký lidskému myšlení a umožňuje zachycení (pro určitý účel) podstatné části „reálného světa". Používá se proto zejména při popisu řešeného problému a posléze pro vytvoření zadání programu.

Algoritmizace a programování

Etapy vývoje softwaru - procvičení

V levém sloupci jsou uvedeny jednotlivé činnosti při vytváření programu. Pokuste se (kliknutím myší na jednotlivé položky a jejich přetažením nahoru/dolu se stisknutým levým tlačítkem myši) tyto činnosti seřadit (od shora dolů) tak, jak jsou standardně vykonávány v čase při vývoji nového programu:



Jak s

Ne… již problému rozumíme, pokusíme se vypraco… ní neomylný, a proto po vytvoření programu je vhodné ho pořádně otestovat a odstranit případné problémy.

Pokud již myslíme, že je program dostatečně funkční, dodáme ho uživateli. Uživatel bude program používat – někdy je však třeba ho nejprve seznámit se všemi funkcemi programu a způsobem, jakým má program využívat. V průběhu používání programu navíc mohou nastat situace, kdy program bude vyžadovat nové funkčnosti - například účetní programy je třeba aktualizovat při změně sazby DPH – proto dodavatel software provádí často také tzv. údržbu programu.

Jednoduché programy může vytvořit i jedinec, velké programy však tvoří obvykle skupina lidí, kteří jsou specialisti na jednotlivé činnosti.

Absolvováním této kapitoly poznáte, co jsou vývojové diagramy a k čemu slouží. Dokážete se v nich orientovat, i je vytvářet a upevnite si tak znalosti z oblasti algoritmů.

V neposlední řadě zde získáte základní přehled o objektovém přístupu.

Orientační časová náročnost kapitoly: 25 minut

Algoritmizace a programování

Co je vývojový diagram?

Již bylo řečeno, že před začátkem programování je vhodné nejprve přesně popsat nejen co řešíme, ale také jak to řešit (jedna z činností zejména koho?). Velmi srozumitelnou možností jak popsat, co bude počítač provádět, představuje vývojový diagram.

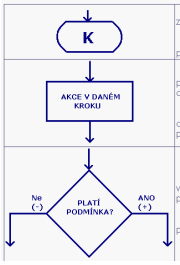
Velmi častým problémem začínajících programátorů je právě skutečnost, že si nejprve dobře nerozmyslí, co vlastně chtějí řešit ani jak přesně bude program pracovat. Pokud je problém složitý, je velmi obtížné uvědomit si všechny souvislosti bez přesného zadání úlohy a kostry postupu řešení.

Vývojový diagram je grafickým vyjádřením algoritmu. Jedná se o posloupnost geometrických obrazců, které jsou propojeny spojnicemi. Které obrazce to mohou být, popisuje následující stránka.

Algoritmizace a programování

Základní značky vývojových diagramů

Pomocí vývojového diagramu lze jednoduchým způsobem popsat algoritmus programu (tedy postup řešení problému počítačem). Základní prvky vývojových diagramů jsou:



Začátek nebo konec algoritmu bude označovat elipsa s vepsaným písmenem (**Z** pro začátek/**K** pro konec).

Pro lepší představu jsou zde u jednotlivých prvků uvedeny šipky. Ty určují, kudy se v programu prochází. Pokud je tento sled jasný, může být šipka vynechána a použije se pouze úsečka.

Jakýkoli příkaz zapíšeme do obdélníku. Pro vývojové diagramy existuje norma, která tento prvek mění a tím upřesňuje, o jakou akci se jedná (například místo obdélníku pro ruční zadání dat). V následujícím textu však budeme využívat pouze tři zde uvedené značky.

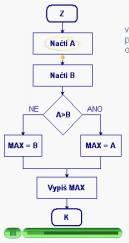
Obdélník může teoreticky obsahovat další skupinu akcí. Ty by poté bylo možno opět zapsat do dalšího vývojového diagramu. Tato skupina příkazů, spustitelná jako celek, se nazývá podprogram. Podprogramům se bude více věnovat závěrečná kapitola kurzu.

„Kouzlo programů" spočívá v možnosti se určitým způsobem rozhodovat. Rozhodování vyznačíme tímto kosočtvercem s právě jedním vstupem a dvěma výstupy, který má uvnitř zapsánu podmínku.

Pokud je podmínka splněna, pokračuje se větví označenou jako **ano** (případně **+**). Není-li podmínka splněna, pokračuje se větví s popiskem **ne** (minus).

Algoritmizace a programování

Vývojový diagram maxima ze dvou čísel



Vlevo je vývojový diagram, který řeší problém nalezení většího ze dvou zadaných čísel. Zřetelně je na něm vidět postup: nejprve jsou zadána dvě čísla (například z klávesnice, načtena ze souboru apod.). Poté jsou porovnána a větší z nich je vloženo do proměnné MAX. Nakonec je tato proměnná vypsána (například na obrazovku počítače).

Animace ukazuje průběh v případě, že A je větší než B (např. A = 10 a B = 2).

Je vhodné si zde připomenout vlastnosti algoritmu, tedy postupu, který je:

- hromadný – tento postup nalezne maximum pro libovolná dvě čísla,
- jednoznačný – v každém kroku je jednoznačné, co bude následovat (podmínku „A>B" považujeme rovněž za jednoznačnou – je-li větší A, bude následovat pravá část, jinak levá),
- opakovatelný – uvedený postup je možné uskutečnit vícekrát,
- konečný – zde je zcela zřetelné, že každá trasa vede od počátku k cíli (K je vždy dosaženo; někdy se však může stát, že se určitá část postupu „donekonečna" opakuje),
- správný - obecně velmi obtížné dokazatelná vlastnost algoritmu. Již zde by mohl být problém například v případě, že jsou obě čísla stejná. (Závisí na konkrétním zadání, zda algoritmus nemá vrátit nějakou jinou hodnotu v případě rovnosti čísel. Naštěstí v tomto případě pokud neplatí podmínka, algoritmus přesto vrátí „největší" hodnotu ze zadaných čísel. Problém správnosti algoritmu může být například dělení dvou čísel – pokud by dělitel byl roven nule.)

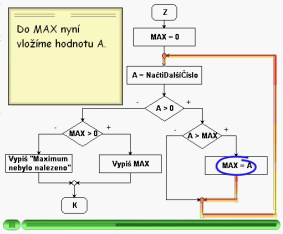
Algoritmizace a programování

Vývojový diagram - maximum

Výpravo uvedený příklad zachycuje obecnější řešení hledání maxima. Jedná se o vyhledání maxima z více čísel. Maximum je zde hledáno pouze z kladných čísel (větší než nula) – čísla jsou postupně zpracována až do chvíle, kdy je zadáno číslo, které není větší než nula. (V tom případě program vypíše maximum z předchozích čísel).

Vývojový diagram obsahuje navíc slučovací bloky (kolečka, do kterých vedou dvě šipky a vychází z nich jedna) – ty jsou často také využívány při zápisu vývojových diagramů.

Zamyslete se, k čemu slouží podmínka MAX > 0 – a kliknete zde pro zobrazení správné odpovědi.



Algoritmizace a programování

OOP

OOP je velmi důležitou zkratkou označující **objektově orientované programování**. Téměř všechny jazyky používané v současnosti umožňují alespoň do určité míry programovat objektově.

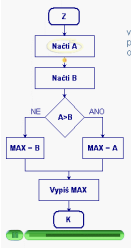
Objektový přístup se snaží přiblížit programování lidskému uvažování. OOP přesahuje rozsah tohoto kurzu, ale uvedeme zde alespoň základní rysy, abychom ho mohli později využít v programování. Přiblížíme si objektový přístup na tzv. tečkové notaci, která je často v OOP využívána. Základním kamenem programování je objekt – například soubor, proměnná. Pokud něco po objektu chceme, využijeme zápis objekt.příkaz (soubor.otevřiSe, obrázek.zvětšiSe apod.). Takto lze postupovat i dále – objekt se může **skládat** z více objektů (cd.prvníSkladba.prehrajSe, clovek.ruka.zvedni).

Pokud po objektu něco chceme, zasíláme mu **zprávu**. Zpráva může mít charakter určitého příkazu – pak hovoříme o zavolání **metody**, nebo se může jednat o stav – to se nazývá **atribut** (příkladem atributu objektu člověk může být jméno, datum narození apod.).

Často se může stát, že ne vše co objekt dokáže, má být přístupné. Tvůrce objektu může určit, na které zprávy bude objekt reagovat - tato vlastnost se nazývá **zapouzdření** (enkapsulace).

Algoritmizace a programování

Vývojový diagram maxima ze dvou čísel



Vlevo je vývojový diagram, který řeší problém nalezení většího ze dvou zadaných čísel. Zřetelně je na něm vidět postup: nejprve jsou zadána dvě čísla (například z klávesnice, načtena ze souboru apod.). Poté jsou porovnána a větší z nich je vloženo do proměnné MAX. Nakonec je tato proměnná vypsána (například na obrazovku počítače).

Animace ukazuje průběh v případě, že A je větší než B (např. A = 10 a B = 2).

Je vhodné si zde připomenout vlastnosti algoritmu, tedy postupu, který je:

- hromadný – tento postup nalezne maximum pro libovolná dvě čísla,
- jednoznačný – v každém kroku je jednoznačné, co bude následovat (podmínku „A>B" považujeme rovněž za jednoznačnou – je-li větší A, bude následovat pravá část, jinak levá),
- opakovatelný – uvedený postup je možné uskutečnit vícekrát,
- konečný – zde je zcela zřetelné, že každá trasa vede od počátku k cíli (K je vždy dosaženo; někdy se však může stát, že se určitá část postupu „donekonečna" opakuje),
- správný - obecně velmi obtížné dokazatelná vlastnost algoritmu. Již zde by mohl být problém například v případě, že jsou obě čísla stejná. (Závisí na konkrétním zadání, zda algoritmus nemá vrátit nějakou jinou hodnotu v případě rovnosti čísel. Naštěstí v tomto případě pokud neplatí podmínka, algoritmus přesto vrátí „největší" hodnotu ze zadaných čísel. Problém správnosti algoritmu může být například dělení dvou čísel – pokud by dělitel byl roven nule.)

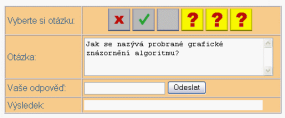
Algoritmizace a programování

Procvičení znalostí

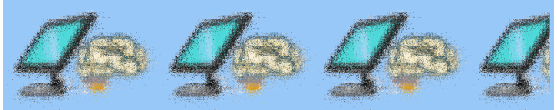
Vyzkoušejte si své znalosti

Vyberte si otázku kliknutím na obrázek otazníku. Poté zapište odpověď a stiskněte tlačítko "Odeslat".

Výsledky tohoto kvízu se neukládají a slouží pouze k procvičení probírané látky.



Visual Basic a makra



Poslední kapitola kurzu je věnována programování. Získáte v ní základní znalosti, dovednosti i návyky potřebné k tvorbě programů v jazyce Visual Basic, zejména pro tvorbu maker.

Orientační časová náročnost kapitoly: 40 minut

Algoritmizace a programování

Úvod do programování

Předchozí text již vytvořil dostatečný teoretický základ, abyste věděli, jak program vzniká a dokázali vytvořit jeho algoritmus. Algoritmus má tu zásadní výhodu, že „je platný obecně", víceméně bez ohledu na konkrétní počítač či programovací jazyk.

Tato poslední kapitola kurzu se věnuje již konkrétnímu programování. Jazyk, který zde poznáte, spadá mezi jazyky vycházející z programovacího jazyka Basic. Jeho velkou výhodou je, že v něm lze programovat programy pro internet, stejně jako třeba vytvářet určitá rozšíření aplikace Word. Právě s oblastí programování v kancelářských aplikacích (MS Office, OpenOffice, 602 Suite) se pravděpodobně nejčastěji setkáte. Navíc se jedná o tematický okruh maturitních otázek.

Vytvořením programu si pak můžete výrazně usnadnit a urychlit práci například s textovým editorem, kdy místo opakovaného zadávání stejných příkazů vytvoříte svůj vlastní program, který tuto sekvenci (posloupnost) příkazů provede za vás.

Algoritmizace a programování

Proměnné a jejich typy

Pokud potřebujeme například sečíst dvě čísla, pak je třeba počítači říct, že si pro ně má vymezit určité místo ve své paměti. **Deklarace proměnné** znamená příkaz počítači, že budeme potřebovat nějakou část jeho paměti, kam si bude ukládat hodnotu určité proměnné. Proměnné budeme deklarovat vždy jejich jménem a tzv. typem. **Typem proměnné** určíme, co může být v proměnné uloženo (například číslo, text nebo datum) i jak je možné s tím pracovat (vydělit dvěma, převést malá písmena na velká, či zjistit rok). Některé typy proměnných v jazyce Visual Basic uvádí tato tabulka:

`Integer`	celá čísla od -32 768 do 32 767.
`Double`	desetinná čísla
`Date`	datum
`String`	řetězec, tedy určitý text (posloupnost znaků).
`Boolean`	tzv. logická proměnná může nabývat hodnot pravda/nepravda (True a False). Využívat ji lze například v různých podmínkách (pokud platí dělej..., jinak dělej...)

Příkaz deklarace pak zapíšeme následovně: `Dim mojeCiselnaPromenna As Integer`. Potom je již možné s touto proměnnou pracovat – například příkaz `mojeCiselnaPromenna = 2 + 2` uloží to této proměnné hodnotu 4. Pokud poté potřebujeme tuto hodnotu vynásobit dvěma, stačí zapsat `mojeCiselnaPromenna = mojeCiselnaPromenna * 2`.

Algoritmizace a programování

Základní znalosti a návyky při vývoji softwaru

Pokud je třeba do proměnné přiřadit hodnotu jiné proměnné, pak stačí zapsat `promenna1 = promenna2`. Jak ale přiřadit text? Základem je jeho zapsání do uvozovek `textovaPromenna = "nějaký text"`.

V programování existují určité návyky, které je vhodné dodržovat. Například zapisování názvů proměnných bez diakritiky (tj. bez háčků a čárek) a rozlišování malých a velkých písmen (case-sensitivita (◐)). Rovněž v Basicu není nutné uvést datový typ proměnné, to však může poté vést k neočekávanému chování, a proto zde budeme datový typ vždy uvádět.

Do programů je vhodné dávat také určité popisky, které sice nemají význam příkazů, ale slouží ke zpřehlednění zdrojového kódu pro programátory. Jako komentář se bere vše, co je zapsáno za apostrofem (`'nějaký komentář`).

Každý příkaz budeme většinou psát na samostatný řádek. Příkazy v příkladech budou mít na začátku řádku vždy určité odsazení (mezery) – takto odsazené texty výrazně usnadňují orientaci, protože související části mají stejné odsazení. Odsazení se proto snažte zachovat i ve svých programech.

Algoritmizace a programování

Procedury a funkce

Procedury a funkce se někdy nazývají **podprogramy**. Představují totiž určitý pojmenovaný blok programu, který pak můžeme vykonat zapsáním jeho jména (hovoříme o jeho **zavolání**). Jedná se o obdobu již probraného rozdílu mezi strojovým kódem (tedy příkazy přímo v „řeči počítače") a vyššími programovacími jazyky. Vyšší programovací jazyky tak vlastně obsahují mnoho podprogramů, které nabízí uživateli jako jednu akci. Uživatel si však může navíc přidat i svůj vlastní podprogram.

Voláme-li podprogram, napíšeme jeho název a za něj závorky. Pokud chceme podprogramu předat nějaké proměnné, uvedeme jejich seznam do těchto závorek – např. `mojeFunkce1(promenna1, promenna2)`.

Rozdíl mezi procedurou a funkcí je v tzv. **návratové hodnotě**. Procedura žádnou návratovou hodnotu neobsahuje, prostě vykoná určitou sekvenci příkazů. Funkce také vykoná příkazy, ale navíc vrátí jednu hodnotu, kterou je možné přiřadit do proměnné. Například `mojeCiselnaPromenna = mojeFunkce2()` přiřazuje návratovou hodnotu funkce do proměnné `mojeCiselnaPromenna`. Jak vytvořit novou funkci či proceduru je uvedeno níže.

```
Function mojeFunkce()
   'případné příkazy funkce
   'vracející vždy 1
   mojeFunkce = 1
End Function
```

```
Sub mojeProcedura
   'případné příkazy procedury
End Sub
```

Návratovou hodnotu tedy uvnitř funkce přiřadíme názvu této funkce.

Algoritmizace a programování

Příklad na podprogramy

Zkuste se na základě stávajících znalostí zamyslet, který podprogram vrátí jako svůj výsledek obsah kruhu.

```
Sub vypoctiObsahKruhu(polomer As Integer)
    ' tenhle podprogram je určitě správně :o)
    vypoctiObsahKruhu = 3.14 * polomer * polomer
End Sub
```

```
Function vypoctiObsahKruhu(polomer As Integer)
    ' tenhle podprogram je určitě správně :o)
    vypoctiObsahKruhu = 3.14 * polomer * polomer
End Function
```

```
Sub vypoctiObsahKruhu(polomer As Integer)
    ' tenhle podprogram je určitě správně :o)
    vysledek = polomer * polomer
End Sub
```

```
Function vypoctiObsahKruhu(polomer As Integer)
    ' tenhle podprogram je určitě správně :o)
    vysledek = 3.14 * polomer * polomer
End Function
```

Zkontrolovat moji volbu...

Algoritmizace a programování

Co je to makro?

Až doposud se v textu předpokládalo, že výsledkem programování bude nějaký spustitelný program. K vytváření programu, který je přímo spustitelný je vhodné mít určitý software, který proces napsání programu v programovacím jazyce usnadňuje i znát jak se ovládá.

Existuje však také jednodušší a snadno dostupná možnost procvičení psaní programu. Jedná se o **makra**, které je možno vytvářet například v běžných kancelářských programech (Wordu, Excelu apod.).

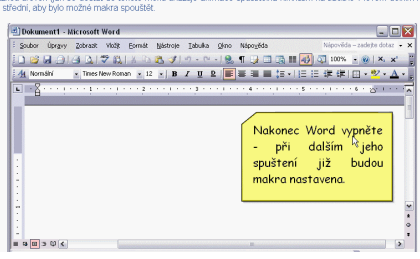
Makro je v tomto pojetí určitá posloupnost příkazů, které se vykonávají v rámci kancelářského programu. Spuštěním makra se vykonají příkazy, tak jak byly naprogramovány.

Makro jazyk využívaný v produktech firmy MS má svůj základ v jazyce Visual Basic a nazývá se **Visual Basic for Application** (**VBA**). Na následujících stranách se naučíte jak v tomto jazyce vytvářet své vlastní programy (přesněji řečeno svá vlastní makra).

Algoritmizace a programování

Makra a úroveň zabezpečení

Protože makra mohou obsahovat také programy, které škodí uživatelům počítače (například mažou určité soubory), mohou být vypnuta. Než se tedy pustíte do programování, zkontrolujte, že Váš Word umožňuje jejich spuštění. To lze nastavit v menu *Nástroje\Makro\Zabezpečení...*, kde lze zadat různou úroveň (jak se dostat do tohoto menu ukazuje animace spustitelná kliknutím na tlačítko v levém dolním rohu). Tuto úroveň nyní nastavte na střední, aby bylo možné makra spouštět.



Algoritmizace a programování

První program

Pro začátek si vyzkoušíme zobrazit uživateli text *Ahoj*. Nejprve si tedy spusťte Váš textový editor a v něm otevřete nový dokument, který uložte pod libovolným názvem. Přes menu *Nástroje\Makro\Makra* se dostanete k již existujícím makrům. Budeme vytvářet nové makro, proto zadejte jeho název (AhojSvete) a poté stiskněte *Vytvořit*. (Výběr *Přehledat* určuje, kde se makro nachází (v případě vytvoření, kde se nacházet bude) – makra mohou být pro všechny dokumenty, nebo pouze pro aktuální.)

Algoritmus to bude velmi jednoduchý – je zde jediná akce, která zobrazí text předaný jako parametr v novém okně (zavoláme podprogram `MsgBox` s parametrem textu, který chceme zobrazit). Po vytvoření makra a jeho uložení, je ho možné ve stejném menu i spustit.


```
Sub AhojSvete()
'
' AhojSvete Makro
' Vytvořeno 1.1.2011 velkým programátorem Ondrou
'
    MsgBox ("Ahoj")
End Sub
```

Animaci vytvoření je možné shlédnout zde.

Algoritmizace a programování

Procvičení vytvoření programu

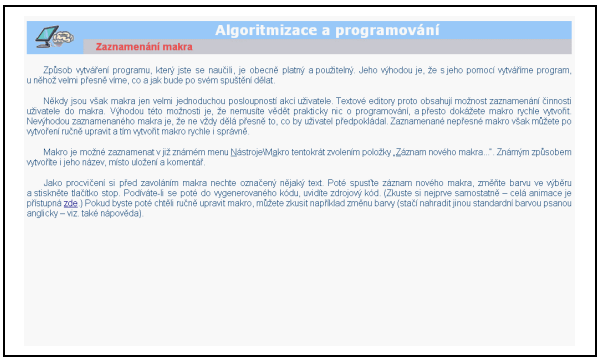
Nyní vytvoříme makro, které bude sčítat čísla zadávaná uživatelem až do chvíle, dokud nezadá číslo 0. (Uživatel bude nejprve vyzván, aby zadal číslo. Poté co zadá poslední číslo (nulu), bude mu zobrazena informace o počtu zadaných čísel a jejich součtu.)

Sestrojte si (nakreslete třeba na papír) svůj vlastní návrh vývojového diagramu a poté se podívejte na možný návrh zde.

Již znáte způsob vytváření makra, proto si přečtěte o dalších podprogramech v tomto odstavci a pokuste se vytvořit program. Jsou využity následující podprogramy:

- funkce `InputBox(text)` – zobrazí (vypíše) proměnnou text a návratovou hodnotou je vstup zadaný uživatelem,
- `Do ... Loop While(podminka)` – pokud je podmínka pravdivá, dojde k návratu na `Do`, jinak se pokračuje v programu ,
- `If (podminka) Then ... Else ... End If` – pokud platí podmínka, pokračuj až do Else (a větev Else se nevykoná), pokud neplatí, pak vykonej pouze větev Else ,
- `Str()` – jedná se o přetypování libovolného typu na `String`. (Pokud je proměnná nějakého typu, nelze do ní libovolně přiřazovat jiný typ. V našem případě funkce vrací `String` namísto `Integer`.)

Výsledný zdrojový kód bude zobrazen po kliknutí zde.



Algoritmizace a programování

Zaznamenání makra

Způsob vytváření programu, který jste se naučili, je obecně platný a použitelný. Jeho výhodou je, že s jeho pomocí vytváříme program, u něhož velmi přesně víme, co a jak bude po svém spuštění dělat.

Někdy jsou však makra jen velmi jednoduchou posloupností akcí uživatele. Textové editory proto obsahují možnost zaznamenání činnosti uživatele do makra. Výhodou této možnosti je, že nemusíte vědět prakticky nic o programování, a přesto dokážete makro rychle vytvořit. Nevýhodou zaznamenaného makra je, že ne vždy dělá přesně to, co by uživatel předpokládal. Zaznamenané nepřesné makro však můžete po vytvoření ručně upravit a tím vytvořit makro rychle i správně.

Makro je možné zaznamenat v již známém menu Nástroje\Makro tentokrát zvolením položky „Záznam nového makra...". Známým způsobem vytvoříte i jeho název, místo uložení a komentář.

Jako procvičení si před zavoláním makra nechte označený nějaký text. Poté spusťte záznam nového makra, změňte barvu ve výběru a stiskněte tlačítko stop. Podíváte-li se poté do vygenerovaného kódu, uvidíte zdrojový kód. (Zkuste si nejprve samostatně – celá animace je přístupná zde.) Pokud byste poté chtěli ručně upravit makro, můžete zkusit například změnu barvy (stačí nahradit jinou standardní barvou psanou anglicky – viz. také nápověda).

Algoritmizace a programování

Otestujte si své znalosti

Následující otázky slouží pro Vaši kontrolu, jak přesně jste pochopili teoretickou stránku obsahu kurzu. Pokuste se vybrat vždy odpověď, která nejvíce odpovídá zadání.

1) Strojový kód je:

- zápis programu v programovacím jazyce (například Visual Basicu)
- heslo nutné ke spuštění počítače
- zápis programu v „jazyku počítače"
- operační systém

2) Přiřaďte jednotlivé činnosti k profesím:

zjišťuje co a jak se bude řešit	Analytik
zapisuje příkazy v programovacím jazyce	Analytik
hledá chyby programu	Analytik
vysvětluje uživatelům, jak mají program používat	Analytik

3) Myšlenková mapa obvykle:

- je uložena na disku počítače
- je uložena na přenosných médiích (disketa, CD, DVD)
- předepisuje testerům, jak mají testovat
- pomocí slov a obrázků popisuje určitý problém

4) Pokud máme objekt auto, který má vlastnost barva s hodnotou modrá, jedná se o:

- skládání
- zprávu
- metodu
- atribut

5) Skutečnost, že objekt může obsahovat další objekty (například objekt byt objekty místnosti) se nazývá:

- zapouzdření (enkapsulace)
- skládání
- nakládání
- řízení tokem událostí

6) Integer je typ proměnné sloužící k uložení:

- textu
- celého čísla
- času
- logické proměnné (true/false)

7) Návratová hodnota je vrácena jako výsledek:

- algoritmu
- funkce
- procedury
- myšlenkové mapy

Stiskněte tlačítko pro zkontrolování odpovědí

Algoritmizace a programování

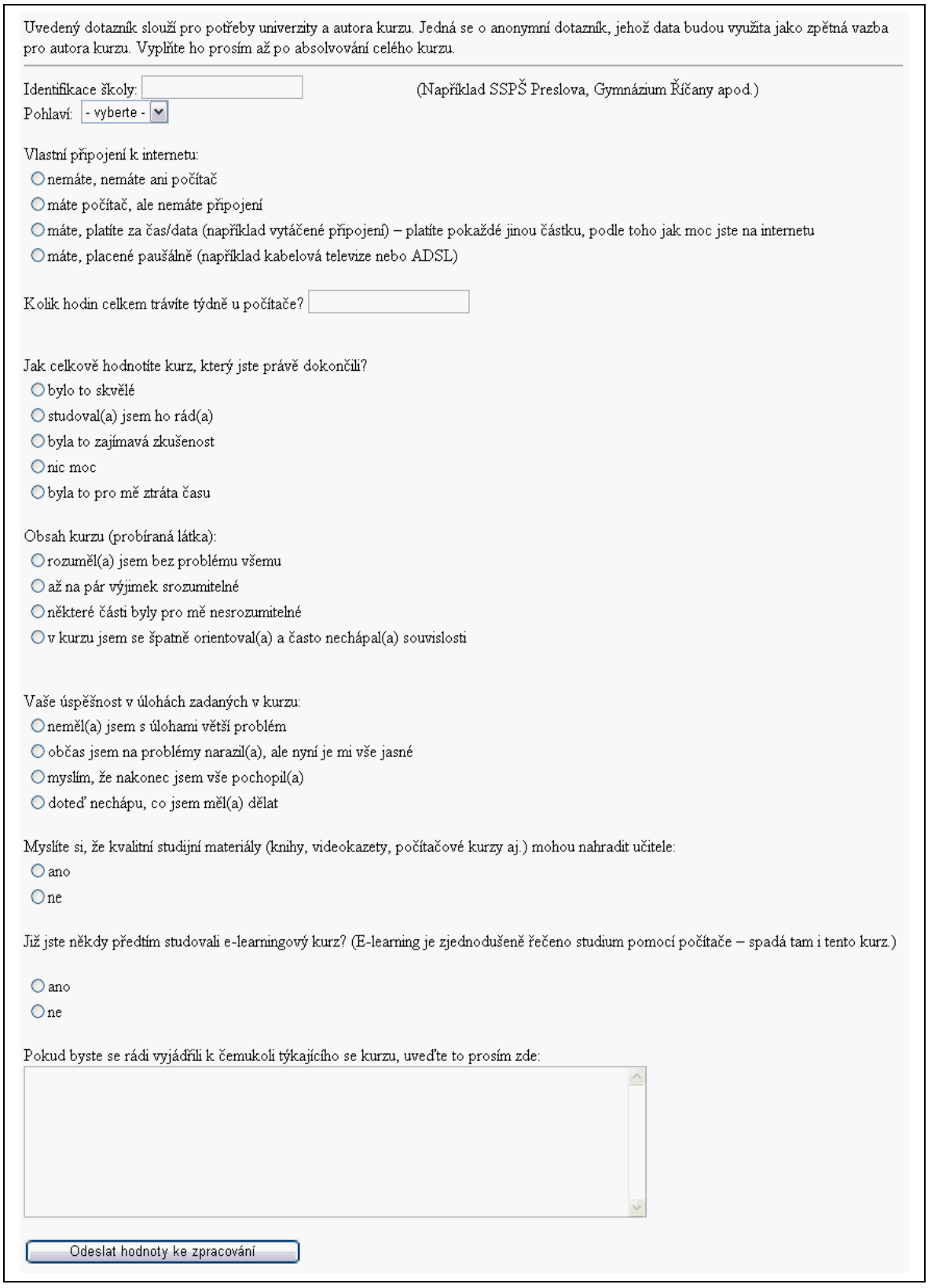
Další zdroje informací

V kurzu, který jste právě absolvovali, jste se dozvěděli základy toho, jak programy fungují a jak se vytvářejí. Pokud Vás programování zaujalo, lze doporučit ke studiu například následující zdroje:

- Popis VBA - v češtině napsaný materiál zabývající se příkazy jazyka Basic,
- Zdroje na stránkách společnosti Microsoft .NET umožňuje vytvářet programy pro počítače i internet v různých jazycích včetně Visual Basicu. Na stránkách společnosti Microsoft naleznete vývojové prostředí zdarma, návody k vytváření programů v češtině i mnoho dalších informací.
- Úvod do problematiky vlastních maker
- Makra v OpenOffice.org

Dotazník ke kurzu

Vyplňovaný dotazník v elektronické podobě je přístupný na stránkách http://algoritmizace.asp2.cz/Test.aspx, jednotlivé vyplněné odpovědi pak prostřednictvím http://algoritmizace.asp2.cz/Test_vysledky.aspx.



Uvedený dotazník slouží pro potřeby univerzity a autora kurzu. Jedná se o anonymní dotazník, jehož data budou využita jako zpětná vazba pro autora kurzu. Vyplňte ho prosím až po absolvování celého kurzu.

Identifikace školy: (Například SSPŠ Preslova, Gymnázium Říčany apod.)

Pohlaví: - vyberte -

Vlastní připojení k internetu:
- nemáte, nemáte ani počítač
- máte počítač, ale nemáte připojení
- máte, platíte za čas/data (například vytáčené připojení) – platíte pokaždé jinou částku, podle toho jak moc jste na internetu
- máte, placené paušálně (například kabelová televize nebo ADSL)

Kolik hodin celkem trávíte týdně u počítače?

Jak celkově hodnotíte kurz, který jste právě dokončili?
- bylo to skvělé
- studoval(a) jsem ho rád(a)
- byla to zajímavá zkušenost
- nic moc
- byla to pro mě ztráta času

Obsah kurzu (probíraná látka):
- rozuměl(a) jsem bez problému všemu
- až na pár výjimek srozumitelné
- některé části byly pro mě nesrozumitelné
- v kurzu jsem se špatně orientoval(a) a často nechápal(a) souvislosti

Vaše úspěšnost v úlohách zadaných v kurzu:
- neměl(a) jsem s úlohami větší problém
- občas jsem na problémy narazil(a), ale nyní je mi vše jasné
- myslím, že nakonec jsem vše pochopil(a)
- doteď nechápu, co jsem měl(a) dělat

Myslíte si, že kvalitní studijní materiály (knihy, videokazety, počítačové kurzy aj.) mohou nahradit učitele:
- ano
- ne

Již jste někdy předtím studovali e-learningový kurz? (E-learning je zjednodušeně řečeno studium pomocí počítače – spadá tam i tento kurz.)
- ano
- ne

Pokud byste se rádi vyjádřili k čemukoli týkajícího se kurzu, uveďte to prosím zde:

Odeslat hodnoty ke zpracování