LITERATE PROGRAMMING NA STŘEDNÍ ŠKOLE

Ladislav Kašpárek
Střední průmyslová škola Jihlava, kasparek@sps-jia.cz

ANNOTATION:
"Literate" programs, their documentation and whole philosophy of Literate Programming could have more dimension: the possible usage of LP at both secondary schools and at schools for higher education (colleges, universities), especially in subjects dealing with programming techniques, algorithms, etc. It is therefore essential to devise a new educational technique based on Literate Programming and to attest it in the real process of teaching and learning. The new educational technique based on LP and possibilities how to use these programs and their documentation during lessons and selfstudying are often discussed.

ABSTRAKT:
Programy a dokumentace, které vznikly v souladu s Literate Programming mohou mít několik rozměrů použití: jedním ze způsobů použití pro výuku programování a algoritmů, zejména na středních a vysokých školách. Proto je nutné navrhnout a v praxi ověřit novou vyučovací metodu, založenou na LP. Příspěvek rozebírá různé možnosti použití LP dokumentů ve vyučování a samostudia na střední škole. Článek stručně seznamuje s principy Literate Programming a možnostmi jak využít programy napsané v systému WEB pro výuku programování na středních školách. Jsou diskutovány možnosti jak použít tyto programy a dokumentace k nim při výuce ve vyučovacích hodinách i při samostudiu.

KLÍČOVÁ SLOVA:
literate programming, výuka, metoda programování, střední škola, TeX

Literate programming

Literate Programming obecně je způsob zápisu programu, kde se do jednoho zdrojového souboru zapisuje program (zdrojový text v programovacím jazyce) a dokumentace k němu. Soubor je rozdělen do sekcí a každá sekce se skládá právě z dokumentace a části kódu, která tvoří relativně samostatný a logický celek programu. Z jiného úhlu pohledu můžeme také říci, že Literate Programming je způsob programování (vytváření algoritmů, jejich zápis v programovacím jazyce) a jeho současné dokumentování – lze je tedy chápat i jako proces vzniku programu. Autorem myšlenky Literate Programming je prof. Donald E. Knuth, který také vyvinul jeho první implementaci – systém WEB [2,3].

Literate Programming se do češtiny volně překládá jako „dobře dokumentované programy“ [4] nebo „kultivované programování“. Ani jeden z překladů nepovažuji za vhodný, protože zní nepřirozeně a podstatu věci nevystihuje přesně, proto je dále v textu používán anglický originál.

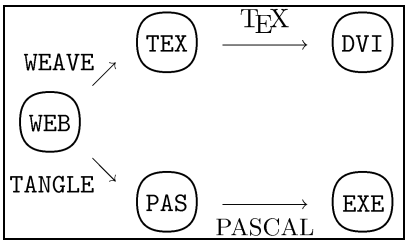
Systémů realizujících Literate Programming je celá řada, např. WEB, CWEB, spiderweb, nuweb atd. Prvním takovým systémem byl již zmíněný WEB, který má jako programovací jazyk Pascal a pro formátování dokumentace používá TeX. Jeden z nejpoužívanějších systémů je CWEB, založený na jazycích C/C++ a systému TeX. Existuje mnoho systémů pro různé programovací jazyky a pro různé formátovací nástroje (nejčastěji TeX) [6]. V současnosti se začínají prosazovat i systémy založené na XML technologiích.

Systém WEB

Systém WEB se skládá z programu tangle, programu weave a formátovacího makra webmac.tex pro TeX. Programátor zapisuje svůj zdrojový text a dokumentaci k němu pomocí řídících kódů do sekcí – vše do jednoho textového souboru[1] s příponou *.web. Pro základní informaci stačí vědět, že pomocí názvů sekcí lze zdrojový kód programu různě slučovat, skládat, případně opakovat v pořadí, ve kterém to vyhovuje programátorovi. S takto vzniklým souborem lze pracovat dvojím způsobem (viz obrázek 1).

Program tangle z web souboru vygeneruje pascalovský zdrojový text a ten lze kompilátorem přeložit.

Program weave z web souboru vygeneruje dokumentaci k programu. Dokumentace je složena z textů jednotlivých sekcí a „naformátovaného“ zdrojového textu v Pascalu. Výsledek zpracuje TeX do finální podoby (dvi, pdf, ps apod.).



Obrázek 1: Práce s WEB souborem.

Výsledkem jsou tedy dvě věci, soubor s programem v Pascalu a precizně naformátovaná dokumentace k programu, včetně rejstříku použitých identifikátorů, rejstříku sekcí, křížových referencí a obsahu.

Tímto způsobem prof. Knuth vytvořil např. programy TeX, Metafont, weave, tangle, StandfordBase a mnoho dalších. Systém CWEB pracuje stejně, jen programovací jazyk je C/C++.

Literate Programming ve výuce

Pro didaktické účely má Literate Programming, a tedy jeho instance systém WEB, několik výhod. Největší výhoda spočívá v tom, že lze zdrojový kód (deklarační a příkazová část) programu rozdělit na relativně malé a samostatné úseky – sekce, včetně jejich komentáře a ty dle potřeby skládat a vytvořit tak celý program. Toho lze také dosáhnout pomocí podprogramů (popřípadě jazyk Pascal umožňuje používání lokálních podprogramů), ale tento způsob má nevýhodu v tom, že vzniká příliš mnoho uměle vytvořených podprogramů a ztěžuje to orientaci ve zdrojovém kódu. Čtení a studování takového kódu je pro studenty obtížné.

Každou sekci lze dokumentovat, tzn. podávat vysvětlení toho, co se v kódu dané sekce děje, tedy co dělají jednotlivé příkazy a co vykonává celá sekce jako celek v kontextu celého algoritmu [1]. Pro učitele to znamená, že se může zaměřit na to podstatné co potřebuje studentům vysvětlit, ať se týká vlastního kódu algoritmu nebo např. deklarací datových struktur.

Výklad může být veden stylem „shora dolů“ (začít přímo hlavním programem, případně hlavičkou podprogramu a „zanořovat se“ do jednotlivých sekcí) nebo může postupovat „zdola nahoru“ (začít jednotlivostmi někde hluboko v sekcích a ty postupně skládat do celku). Obě

[1] Kompletní popis struktury WEB souboru a jeho jednotlivých řídících kódů přesahuje rámec tohoto příspěvku.

dvě možnosti lze samozřejmě vhodně kombinovat. Toto je ideální v hodinách přednáškového typu, kdy lze dokumentaci upravenou a naformátovanou pro dataprojektor promítat, při tom ji komentovat a diskutovat se studenty.

Sekce se mohou do sebe navzájem vkládány tak, aby při jejich vzájemném sestavování nevznikl cyklus. Této vlastnosti lze využít při výkladu tak, že algoritmus (program) je rozdělen do několika „úrovní“ – na obecné sekce a sekce specializované.

Příklad: ssort je program, který autor používá pro výuku řadícího algoritmu select sort. Na obrázku 2 je fragment dokumentace k tomuto programu, na kterém jsou vidět všechny uvedené vlastnosti (např. kód ze sekce 7 je vložen do sekce 6, vše je podrobně popsáno).

6. Abychom algoritmus správně navrhli, musíme si uvědomit, co který cyklus dělá a **kolikrát**!

Kolikrát je potřeba opakovat vyhledání nejmenšího prvku v úseku a pak jej vložit na začátek úseku? Z toho plyne otázka, kolik je úseků? Pro pole délky max je to $(max - 1)$. Zdůvodněte si jako cvičení proč?

Důležitou funkci v algoritmu hraje proměnná *index_min*. Zde je nastavena na začátek řazeného úseku.

⟨ Vnější cyklus algoritmu – řízení průchodů 6 ⟩ ≡

```
for i := 1 to max − 1 do
  begin index_min := i;
  ⟨ Vnitřní cyklus algoritmu – hledání minima 7 ⟩;
  tmp := data[index_min]; data[index_min] := data[i]; data[i] := tmp;
      { záměna obsahu dvou proměnných }
  end;
```

Kód je použit v sekci 3.

7. Vnitřní cyklus vyhledává postupně pozici minima v řazeném úseku. Kolik je k tomu potřeba podmínek – porovnání?

⟨ Vnitřní cyklus algoritmu – hledání minima 7 ⟩ ≡

```
for j := i + 1 to max do
  if data[index_min] > data[j] then index_min := j;
```

Kód je použit v sekci 6.

Obrázek 2: Ukázka dokumentace k programu ssort.

Dokumentace, která je produktem programu weave, se do finální podoby zpracovává systémem TeX, resp. pdfTeX, za použití souboru maker webmac.tex. Pomocí těchto nástrojů lze jednomu dokumentu dát různé podoby – vytisknout program jako brožurku nebo na samostatné listy nebo je také možné vytvořit prezentaci vhodnou pro promítání na dataprojektoru či na interaktivní tabuli. Učitel má možnost připravit takový dokument, který je pro jeho výklad nejvhodnější.

Zmíněné vlastnosti se vztahovaly k dokumentaci. Při výuce lze také používat samotný zdrojový kód programu, tedy produkt programu tangle.

Jsou dvě možnosti, jak zdrojový kód programu ze souboru *.web získat. Pro studenty nepopulární způsob je kód z dokumentace vypsat ručně. Je to sice pracné, ale nutí to studenta o programu, o algoritmu přemýšlet a číst si dokumentaci. Tento způsob lze použít v případě, že program není příliš dlouhý (max. 20–25 řádek textu).

Druhou cestou je použití programu tangle. Výstup originálního programu tangle je primárně určen pro kompilátor, proto jsou z něj vyjmuty všechny nadbytečné mezery, konce řádků a komentáře. Takto zapsaný zdrojový text v jazyce Pascal je pro člověka na první pohled téměř nečitelný. Pokud není kód příliš dlouhý (max. 20 řádů po 80 znacích), studenti

se v něm rychle zorientují (např. podle komentářů, ve kterých jsou čísla sekcí) a jsou schopni jej upravit do čitelné podoby a s tímto zdrojovým textem pak dále pracovat a experimentovat.

Pro delší programy je i toto nepoužitelné, proto je vhodné program tangle upravit tak, aby každý příkaz nebo deklarace byla vypsána na samostatném řádku [5].

Výuka vedená tímto způsobem ukazuje studentovi, jak programátor při vytváření kódu postupoval, jak skládal do sebe jednotlivé kroky algoritmu a jakých programátorských postupů použil pro vyjádření algoritmu v daném programovacím jazyce. Zdrojový text tak dále slouží k experimentování s daným programem (změny vstupních hodnot, konstant apod.) nebo k jeho modifikacím (optimalizacím). Na obrázku 3 je výstup programu tangle, jehož vstupem byl soubor ssort.web.

```
{3:}program ssort;const{4:}max=30;
{:4}var{5:}data:array[1..max]of integer;i,j:1..max;tmp:integer;
indexmin:1..max;{:5}begin writeln('Select sort demo program.');
{8:}randomize;for i:=1 to max do data[i]:=-1000+random(2001);{:8};
{9:}for i:=1 to max do write(data[i]:8);{:9};
{6:}for i:=1 to max-1 do begin indexmin:=i;
{7:}for j:=i+1 to max do if data[indexmin]>data[j]then indexmin:=j;{:7};
tmp:=data[indexmin];data[indexmin]:=data[i];data[i]:=tmp;end;{:6};
writeln;{9:}for i:=1 to max do write(data[i]:8);{:9};end.{:3}
```

Obrázek 3: Výstup programu tangle.

Samostudium

Forma těchto dokumentů je přímo předurčena k samostudiu. Výstup programu weave je v podstatě hypertextovým dokumentem podobně jako html stránky. Ten lze vytisknout jako knihu a také ji tak číst nebo lépe řečeno listovat v ní ze sekce na sekci. Pomocí pdfTeXu je možné vytvořit i interaktivní podobu dokumentu a ten pak pročítat a studovat přímo na počítači.

Doporučení pro psaní výukových materiálů v systému WEB

Tvorba nových ukázkových programů v systému WEB není nijak náročná. Autor, nejlépe zkušený učitel, musí znát základní řídící sekvence systému, musí umět programovat v Pascalu a uživatelsky ovládat plainTeX. Z didaktického hlediska je nejtěžší správně rozložit program do jednotlivých sekcí a dobře je zdokumentovat. K psaní vlastního kódu, jehož velikost nepřesahuje deset stran textu, postačí libovolný jednoduchý textový editor. Pro rozsáhlejší projekty je vhodné použít nějaký sofistikovaný editor např. EMACS, vim nebo pro Windows Leo.

WEB je propracovaný systém pro tvorbu dokumentovaných programů v jazyce Pascal. Např. obsahuje preprocesor podobný tomu, který má jazyk C. Dále má řídící sekvence pro formátování dokumentace a pro manipulaci s výsledným zdrojovým textem. Pro výukové účely je vhodné psát dokumenty co nejjednodušší a tedy některé pokročilé vlastnosti systému nepoužívat. Určitě se nedoporučuje používat preprocesor a jeho makra, protože studenti mají problémy pracovat se samotným jazykem.

Shrnutí

Použití programů napsaných v systému WEB vnáší do výuky programování několik nových prvků. Mezi nejdůležitější patří oživení výuky jinou formou zápisu programu, lepší zapojení studentů do procesu výuky a možnost pohodlného a efektivního samostudia. Dále si studenti

zvykají své programy více a lépe komentovat. Vedlejší efekt je propagace typografického systému TeX mezi studenty.

LITERATURA

[1] Kašpárek L.: *Literate Programming at Secondary School*, 4th International PhD Workshop, Spa Libverda September 2003, Department of Adaptive Systems, UTIA ISBN 80-239-1333-6

[2] Knuth D. E.: *Literate Programming*, Stanford University, CA USA 1992, ISBN 0-937-07380-6

[3] Knuth D. E.: *Literate Programming*, The Computer Journal, 27:97–111, 1984

[4] Olšák P.: *Typografický systém TeX*, 1. vyd. CSTUG 1995, ISBN 80-901950-0-8

[5] Sewell W.: *Weaving a program: literate programming in WEB*, Van Nostrand Reinhold, NY, USA 1989, ISBN 0-442-31946-0

[6] http://www.literateprogramming.com