1. Úvod

Tento text je určen pro studenty úvodního kurzu programování studijního oboru Informatika. Obsahuje úvod do objektové teorie, její realizaci v jazyce Java a popis základních jazykových konstrukcí.

Javu učíme v základním kurzu programování od roku 2000, ze začátku paralelně s Pascalem, od roku 2004 výhradně Javu. Do roku 2004 jsme první semestr učili studenty programovat v Javě procedurálně a až druhý semestr jsme se snažili vysvětlit objekty. Naráželi jsme však na následující problémy:

- Část studentů v úvodním kurzu již měla předchozí znalosti programování – obvykle Pascalu či PHP. Vzhledem ke svým předchozím zkušenostem s programováním a algoritmizací se tito studenti na úvodních cvičeních nudili a nemuseli se programování věnovat mimo cvičení. Pak se jim často stávalo, že „zaspali“ okamžik, kdy by se měli začít učit. Vytvářelo to i špatnou atmosféru pro začínající studenty – ti viděli, že někteří jejich kolegové nemají žádné problémy s programy a byli z toho zbytečně frustrovaní.
- Mezi algoritmickým myšlením a chápáním objektů je poměrně velký myšlenkový rozdíl. Mnoho studentů nebylo schopno v rámci druhého semestru přejít od algoritmického myšlení k objektovému. Nechápali, proč mají používat objekty, když předtím používali objektový jazyk neobjektově. Vzhledem k rozsahu jimi laděných programů je obtížné ukázat jim výhody objektového přístupu pro řešení složitých úloh.

V Computing Curricula [CC2001] z roku 2001 se doporučuje v úvodních kurzech programování začít objektovým přístupem, tj. od začátku výuky programování soustředit pozornost na objektově-orientované programování a návrh. Za základní výhodu tohoto přístupu je označováno brzké seznámení s objektově orientovaným programováním, které je standardem při tvorbě programového vybavení. Objektový přístup je výhodný pro složitější úlohy, tudíž studenti se při výuce budou od začátku seznamovat se složitějšími aplikacemi, které mají složitější vnitřní vazby.

V rámci katedry jsme se rozhodli přejít při výuce programování k přístupu object-first. V úvodním kurzu tedy začínáme výukou objektů, v navazujícím kurzu Základy softwarového inženýrství si pak studenti tyto znalosti prohloubí a seznámí se i s návrhovými vzory, testováním tříd atd.

Objekty byly navrženy pro pochopení/zvládnutí složitých problémů. Studentům je potřeba ukazovat používání objektů na „složitých“ problémech. Přitom je potřeba výuku připravit tak, aby studenti mohli postupovat od jednoduchého ke složitějšímu. V naší výuce objektů jsme se inspirovali přístupem M. Köllinga [Kolling], autora výukového vývojového prostředí BlueJ a propagátora přístupu k výuce object-first. Rozhodli jsme se pro následující postup ve výuce:

- nejdříve ukážeme studentům na existující aplikaci, jak fungují objekty, jaký je rozdíl mezi třídou a instancí (vytvořit více instancí), jak volat metody instancí,
- na jiné úloze studenti mění hodnoty řetězců, naučí se na tom překládat a rozumět chybovým hlášením překladače,
- dále následuje úprava obsahu metod či psaní obsahu metod,
- dalším krokem je návrh metod v existujících třídách,
- dále studenti k existujícím třídám navrhují další třídy,
- posledním krokem je návrh aplikace od začátku do konce.

Na přednáškách vykládáme objekty a jazyk, na cvičeních se řeší problémy. Cílem cvičení je trénování objektového myšlení na řešení předložených problémů, tj. ukázat jim problémy, diskutovat s nimi jak je řešit.

Tomuto členění odpovídá i rozdělení tohoto textu do dvou částí. První část (kapitoly 2 – 13) obsahuje výklad jazykových konstrukcí. Zdrojové kódy v těchto kapitolách jsou většinou krátké ukázky, nejsou u nich uvedeny komentáře, protože jsou podrobně vysvětleny v textu. Jednotlivé kapitoly jsou řazeny tak, aby se postupovalo od základních vlastností a prvků ke složitějším.

V druhé části (kapitoly 14 – 20) jsou popsány některé výukové projekty používané na cvičení. Zde jsou uvedeny celé zdrojové kódy, včetně komentářů. Studenti by se měli mimo jiné naučit rozumět cizímu zdrojovému kódu. Každý projekt končí zadáním domácích úkolů, které mají sloužit jako inspirace pro samostatnou práci studentů. I tyto projekty jsou řazeny od jednodušších ke složitějším.

Tato skripta, stejně jako kurz pro který jsou určena, si nekladou za cíl naučit studenty všechny jazykové konstrukce a možnosti Javy. Z těchto důvodů ve skriptech nejsou popsány některé jazykové konstrukce z jazyka Java:

- vnitřní třídy a anonymní vnitřní třídy,
- generické typy (ve skriptech je popsáno pouze jejich používání),
- anotace,
- serializace objektů,
- problematika vláken a synchronizace.

Nejsou zde uvedena pravidla pro navrhování tříd a návrhové vzory. Nejsou zde mnohé standardní knihovny Javy – např. Swing a AWT, pomocí kterých se vytvářejí grafické aplikace, dále práce s databázemi, komunikace po síti, vytváření webových aplikací atd.

Závěrem bychom chtěli poděkovat za cenné připomínky Ing. Martině Jandové, Ing. Rudolfu Pecinovskému, CSc. a za recenzní posudky Ing. Aleně Buchalcevové, PhD. a Prof. Janu Štelovskému.

Doufáme, že tato skripta pomohou čtenářům při pronikání do tajů objektově orientovaného programování a programovacího jazyka Java.

autoři