文章编号：1672-5913(2008)08-0024-03

编译原理课程实践改革探索

张 昱，陈意云

(中国科学技术大学 计算机科学技术系，合肥 230026)

摘 要：本文介绍了我系对编译原理课程实践的改革，陈述了该实践活动的内容、方法、效果和经验教训。

关键词：编译原理；课程实践；改革

中图分类号：G642 **文献标识码：**B

1 引言

当今本科生人数大幅增加，高校毕业生就业竞争加剧，用人单位对人才要求不断提高，计算机及相关专业的不少毕业生在就业过程中暴露出动手能力差、分析问题解决问题能力薄弱、创新意识不强等问题。这些问题的出现在很大程度上反映出高校在学科的专业实践(特别是课程实践)教学方面的不足：

1) 各课程的课程实践各自独立，实践内容跟不上计算机科学与技术的发展，内容陈旧、覆盖面窄、综合性不高、难度低、规模小，不注重对学生工程、质量、团队等意识的培养；

2) 学生数与助教数比率增加，一些学校采用研究生作为助教，助教对学生实验的检查力度和深度不够，难以真实反映学生的实验水平；

3) 未结合新形势下学生的特点来规划和组织实践，学生的热情不高，拷贝风气日益蔓延。现在的学生兴趣广泛，精力分散，多数有计算机，但是投在课程学习及实践的时间大大减少；不少学生学习目标不明确，遇到挫折容易退缩，在学习上的钻劲和毅力有所降低。

针对这种现状，笔者认为加强和改善专业实践应首先抓课程实践改革，而课程实践改革则应以整体规划各计算机专业课的课程实践为指导思想。专业实践所能覆盖的程度依赖于制度的保证、学科机构的资源以及教职人员的利益。

就软件类的课程而言，课程实践主要围绕着软件的设计与实现展开。课程实践的整体目标是学生至少能参与完成一个有一定规模的软件项目的设计与开发，这样的项目应能涉及到对多门课程所学原理的综合运用。在整体规划课程实践时，应遵循由小到大、循序渐进的原则，注意整体规划课程实践所涉及的语言、工具和环境，注意学生软件工程意识、质量意识和团队意识等的培养。

在内容选取上，低年级的课程实践(如 C 语言、数据结构)以巩固课程知识的小实验为主，训练学生基本的程序设计技能；而高年级的课程实践(如编译原理、操作系统等)则应以综合运用的课程设计为主，训练学生软件工程的能力。

在上述思想的指导下，笔者经过两年多的调研和准备，于 2007 年上半年在本系 2004 级学生的编译原理教学实践中开展了编译原理课程实践改革。本文将在以下各节依次介绍这次课程实践改革的内容、方法和实施效果，总结实践中的经验教训，供同行参考。

2 课程实践方案

2.1 课程实践的规划及历程

根据上述指导思想，我们将编译课程实践定位为综合运用的课程设计，即学生(通过合作)为某个实用语言设计和开发一个可运行的编译器。这不仅能使学生加深对编译原理和技术的理解，还能提高学生的软件开发水平。学生在实践中将熟悉和掌握一些软件工程工具、环境和规范，培养工程、质量和团队等意识。

制定这样的课程设计方案，首先要合理选择编译知识点，定义待实现的语言；然后对语言的编译器进行模块划分和预实现，估计实现的难度和工作量；最后研制提供给学生的支持库、样例、工具和文档，明确学生的任务。在方案研制中，既要注意使课程设计有一定的规模，又要考虑到学生和课时的实际情况，以使学生在有限的时间内尽可能多地掌握编译知识并得到综合训练。

为此，我们于 2004 年秋开始调研国外一些知名大学的编译课程设计，从中选择美国加州大学伯克利分校的编译课程设计进行深入分析与研究。我们以本科毕业论文的形式让学生做其中的部分实验，从中感受和总结实验的难

收稿日期：2008-3-12

© 1994-2008 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

度、难点以及工作量等。2006 年起，我们着手设计适合国情的课程设计，它由一系列的小课程设计组成，学生通过循序渐进地做其中的一部分即可实现一个实用语言。我们选取 Java 语言的一个子集 MiniJOOL 作为实验语言，它不支持 import 和 package 指令，也不支持 interface、抽象类和抽象方法、public 等访问控制修饰和异常等，程序中所有的类都放在同一个文件中。这样的语言既具有相当规模的语言特征，又比 Java 语言小得多。但是即便如此，实现这样的语言也不容易。为循序渐进地引导学生进行语言的实现，我们又对 MiniJOOL 进行裁剪，定义了 SimpleMiniJOOL 和 SkipOOMiniJOOL 两个非面向对象语言。前者只允许程序中包含一个方法，后者则包含 MiniJOOL 的所有非面向对象特性。目前，系列课程设计及支持库等仍在不断改进之中，感兴趣的同行可以从 http://staff.ustc.edu.cn/~yuzhang/compiler 获得已研制并已在使用的相关课程实践资源。

为检验系列课程设计及相关资源的合理性和效果，发现其中的疏漏和不足之处，我们在 2007 年上半年的编译原理教学中开展了如 2.2 节描述的编译课程实践，并制定了如 2.3 节描述的考评方法来督促、激励、评价学生的课程设计。

2.2 课程实践的任务

在这次课程实践中，要求学生用 Java 实现 SkipOOMiniJOOL 语言，每个学生需要独立完成编译器的前端(或后端)，并自行选择完成后端(或前端)的合作伙伴。前端要求完成词法分析、语法分析、语义检查并生成抽象语法树(AST)；后端则要求由 AST 生成 x86 汇编码，不要求代码优化，生成的汇编码应能直接用 gcc 汇编连接得到可执行文件。前后端的学生需要定义好接口，不开放源代码给对方，而只提供 jar 文件和接口说明，在运行时应能输出前端和后端的作者名。

我们规定采用 Eclipse JDT(Java Development Tools)中的 AST 实现，但不限制学生实现前端或后端所采用的方法。我们在 3 月底将已编写的课程设计讲义印发给学生，并将相关的工具和支持库等发布在主页上，供学生参考。下面简述本次课程设计涉及的语言、工具、支持库和样例。

1) SkipOOMiniJOOL 语言

一个合法的 SkipOOMiniJOOL 程序有且仅有一个名为 Program 的类定义；类中所有数据和方法成员都必须是 static 的；除主函数外，其余函数都允许有参数和返回值，所有函数都必须由 return 语句返回。出现在程序中的类型只能有 int、boolean、String 以及一维的 int 型数组(数组长度是常量)。

2) 工具

我们选择一些开源的免费工具供学生使用，包括 Java 集成开发环境 Eclipse (http://www.eclipse.org/)、Java SDK(http://java.sun.com)、Ant 编译工具(http://ant.apache.org，编译文件是 XML 格式，比 GNU make 的 makefile 更清晰易懂)、词法分析器的生成工具 JFlex (http://jflex.de/)、语法分析器的生成工具 CUP(http://www2.cs.tum.edu/projects/cup/，支持 LALR(1)文法)或 JavaCC (https://javacc.dev.java.net/，支持 LL(k)文法)、GCC(http://gcc.gnu.org，Windows 下可以用 MinGWStudio (http://www.mingw.org/))。

3) 支持库和样例

讲义中简要描述了 AST 并列出要用到的 AST 节点类，提供 AST 的树型显示类 ASTViewer 便于学生显示 AST。我们以赋值语句序列语言为例，说明如何手工或利用工具构造词法、语法分析器，得到语法树，并提供相关的文法文件、Java 源代码框架和 ant 编译文件等。我们引入访问者模式，并以此为基础提供对 AST 的解释器、语义检查和 x86 代码生成等的代码框架。不过，语义检查和代码生成部分的讲义和代码框架还显得非常粗糙，有待完善。

2.3 考评机制

课程实践的效果不仅取决于实践的内容，还取决于实践中的激励、过程管理和考评机制等。为调动学生的积极性，我们将竞争机制引入到实践中，学生可以自行推销和选择前端(后端)，如果某个学生的前端(后端)被采用得越多，则得分越高。我们在 4 月中旬和 5 月安排两次课堂辅导，并利用校 bbs 的 CompilerTech 版(http://fbbs.ustc.edu.cn/cgi/bbsdoc?board=CompilerTech)和 E-mail 等进行日常交流。

在考评方面，我们将学生分成 15 组，每组约 10 人。每组用近 4 小时的时间进行现场测试、答辩和评分；评委由教师、助教和同组的所有同学担任，教师主导测评过程、学生现场操作并采用投影仪显示；所有评委均可以提问，学生需当众回答，所提问题主要围绕其完成的设计和编程以及测试中暴露出的错误等展开。评委的评分依据主要包括编译器的正确性、错误定位与恢复能力、生成的目标代码质量、回答问题时所表现出的对本课程设计所涉及知识的掌握程度、对自己的前端(后端)的熟悉程度、操作的熟练程度、提交物的完整性和条理性及其中反映的分析和设计思想等。每个评委当场给该组的全部同学排名；由助教根据各有效排名表给出最终排名；由教师根据本组情况确定本组的最高分和最低分，并依据排名确定每个同学的分数。此外，还规定了其他一些评分细则。

3 实施效果与经验教训

在这次课程实践中，一些学生的积极性被充分调动起来。自 2007 年 5 月 11 日起的一个半月中，学生在校 bbs 的 CompilerTech 版发了约 300 封帖子讨论课程实践，改变了该版自2005年11月开版以来不太活跃的状况(该版自开版到 2007 年底的总贴数仅为 978 封)。十来个学生编写的前端或后端有较强的语义检查和错误恢复功能，甚至支持一些代码优化功能，部分学生的潜力得到了挖掘。但是，仍有许多学生投入时间不足，采取临时突击的方式，使得结果不好或者没有做完。下面分别总结本次实践的一些经验和教训。

© 1994-2008 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net

3.1 经验

(1) 在所提供的程序框架和文档说明下扩展实现语言的编译器，既有挑战性又有好的效果。实现一个完整的编译器不仅工作量大而且有难度，提供程序框架和文档给学生，让学生先阅读再设计编码，这能使学生易上手并降低难度，不会出现大的设计偏差。在实践效果上，学生不仅能巩固从课本所学的编译器各个阶段的功能和技术，增强实践能力，而且补上了对编译器的整体认识。

(2) 以 AST 为中间结构将实验划分为前后端两类任务，并允许自行设计接口，既控制了学生开发的规模又允许有自行设计的空间。由于规定了 AST 实现，选择前端或后端的学生可以以此为基础分别独立实验；但是，Eclipse JDT 的 AST 实现是面向 Java 语言的，在用它实现 SkipOOMiniJOOL 语言时需要进行适当的扩展，如数组类型的处理、变量的作用域等，这就需要学生自行设计和约定。

(3) 提供 AST Viewer 并要求生成 x86 汇编码，便于测试和考评。有了 AST Viewer，学生和评委可以方便地查看所生成或接收的 AST 是否正确；采用 x86 汇编码，可以利用 gcc 得到可执行文件，从而方便学生和评委测试代码生成的正确性。

(4) 合作开发、自主推销和选择、整体评测，既培养了团队精神，又增强了质量意识。学生虽然只实现前端或后端，但是在评测时要求看整个编译器的优劣，这促使学生相互合作沟通并增强工作责任心。通过自主推销和选择，一些学生积极深入其他宿舍推销产品并承诺和履行售后服务，使学生在实践中建立质量意识，并体会到市场上只接受高质量的或者是提供良好服务的产品。

(5) 规定了统一的版本提交截止时间，既有公平性和工程性，又易于评测。评测同一时间节点的版本，可以避免后评测的学生根据之前的评测情况来完善程序，也可以避免评测开始后不断有新版本来干扰评测。公布截止时间还可以培养学生的工程意识。

(6) 教师主导的集体公开评分方式，既有公平性又易评测。由学生参与评分，既能弥补教师对学生实际情况了解的局限性，又能调动学生的参与热情。尽管存在少数人恶意打分的情况，但是采用记名的排名记分形式，大部分学生的打分都比较公正，恶意打分不起作用。

3.2 教训及改进之处

(1) 为使学生在有限的时间开发出一定规模的编译器并培养工程意识，我们引入了不少开发工具和环境，这加宽了学生的技术层面，但也导致学生不能把精力集中到和编译有关的技术上来。改进的做法是让学生在前导软件课程实践中逐步熟悉掌握其中的部分工具(如 gcc、eclipse 等)，同时提供对使用这些工具的文档说明和样例。

(2) 讲义中对 SkipOOMiniJOOL 语言的描述不够精确，这使得学生对上下文有关的约束不够重视或认识不清；对后端没有较明确的实现要求，所提供的代码生成样例采用逐变量存储分配，并用运算栈完成表达式计算，学生基本上通过修改、扩展该样例完成后端，降低了实验难度。为此，需要进一步形式化语言规范，吸收常规编译器的代码生成做法并改进支持库和样例，细化对后端的实现要求。

(3) 讲义中规定了开发环境目录，但是对提交环境目录和编写能编译运行编译器的批处理文件等要求发布太迟，学生对统一的环境目录、环境设置及批处理文件的编写等没有引起重视，这给评测带来了麻烦。另外各个开发工具都有许多版本，由于没有事先对版本做限定，造成在评测前临时通知并准备多种环境。今后的改进是在讲义中增加对提交环境目录和批处理文件编写指南的描述，说明要考虑哪些版本和环境问题，并给出几种测试环境组合供参考；平时要注意对学生强调这些问题。

(4) 只规定了最后版本的提交截止时间，许多学生采取临时突击的方式，投入时间不足，使得结果不好或者没有做完。对提交内容描述不够细致，缺少过程管理与控制措施，使得学生忽略了环境设置以及使用相对路径和批处理文件等用来保证程序包在其他机器上快速运行的方法；不注重规范，有的学生没有按要求实现语言，更多的人不按要求建立开发环境或进行提交上传；由于没有事先发布一批测试程序，大家对测试关注不够。今后的改进是制定多时间节点和多次提交的过程管理与控制机制，如提交设计文档、提交源代码、发布测试程序、发布评测环境、提交最终版本等；在讲义中细化对开发环境、提交环境、版本问题、批处理文件等的描述，平时反复强调这些事情。

(5) 课程实践要求主讲教师和助教必须熟悉 2.2 节所列的各种工具和环境。助教尤其需要熟悉开发和测试环境等，以便应付学生在实验过程中遇到的问题以及提交不规范所引起的问题等。要教育助教遵守规则，否则截止时间等各种规定变成虚设。对于这样规模的实验，研究生做助教不合适。

4 结束语

这次课程实践让我们看到了少数学生在课程实践中所表现出的才智与个性化特点，也暴露出了许多问题。但是，这些问题对于我们改进系列课程设计，改善支持库、样例以及进一步细化讲义等是大有裨益的；它们也为我们进一步细化过程质量管理细则和考评细则提供了有力的指南。

参考文献

[1] 中国计算机科学与技术教程 2002 研究组. 中国计算机科学与技术学科教程 2002(CCC2002)[M]. 北京:清华大学出版社,2002.

[2] 张昱,陈意云. 编译原理课程设计(草稿)[EB/OL]. http://staff.ustc.edu.cn/~yuzhang/compiler/.2007.

© 1994-2008 China Academic Journal Electronic Publishing House. All rights reserved. http://www.cnki.net