<<编译器构造>>

图书基本信息

书名：<<编译器构造>>

13位ISBN编号：9787302227205

10位ISBN编号：7302227209

出版时间：2010-6

出版时间：清华大学出版社

作者：（美）费希尔，塞特朗，勒布兰　著

页数：683

前言

Much has changed since Crafting a Compiler， by Fischer and LeBlanc， waspublished in 1988. While instructors may remember the 5~-inch floppy disk ofsoftware that accompanied that text， most students today have neither seen norheld such a disk. Many changes have occurred in the programming languagesthat students experience in class and in the marketplace. In 1991 the bookwas available in two forms， with algorithms presented in either C or Ada.While C remains a popular language， Ada has become relatively obscure anddid not achieve its predicted popularity. The C++ language evolved fromC with the addition of object-oriented features. JavaTM was developed as asimpler object-oriented language， gaining popularity because of its securityand ability to be run within a Web browser. The College Board AdvancedPlacement curriculum moved from Pascal to C++ to Java.

<<编译器构造>>

内容概要

本书是一本面向计算机系本科生的编译器教材。
作者在三所美国大学拥有长达25年的编译器教学经验，在本书中对编译器构造的基本知识与关键技术进行了全新的讲解。
本书的主要内容包括：编译器历史和概述、词法分析（扫描）、语法分析（包括自顶向下和自底向上的分析）、语法制导翻译、符号表和声明处理、语义分析、中间表示形式、虚拟机上的代码生成、运行时支持、目标代码生成和程序优化等。
本书提供了详尽清晰的算法，主推在实践中学习编译器构造的相关技术，同时提供了配合教材使用的教学网站、参考资料以及源码下载。
不仅可以作为计算机专业本科生或研究生的参考教材，同时也适合相关领域的软件工程师、系统分析师等作为参考资料。

<<编译器构造>>

书籍目录

章节摘录

An optimizing compiler is specially designed to produce efficient target codeat the cost of increased compiler complexity and possibly increased compila-tion times. In practice， all production-quality compilers （those whose outputwill be used in everyday work） make some effort to generate reasonable targetcode. For example, no add instruction would normally be generated for theexpression i+O. The term optimizing compiler is actually a misnomer. This is because nocompiler of any sophistication can produce optimal code for all programs. Thereason for this is twofold. First， theoretical computer science has shown thateven so simple a question as whether two programs are equivalent is undecid-able： such questions cannot generally be answered by arty computer program.Thus finding the simplest （and most efficient） translation of a program cannotalways be done. Second， many program optimizations require time propor-tional to an exponential function of the size of the program being compiled.Thus， optimal code， even when theoretically possible， is often infeasible inpractice.

版权说明