Design Concepts in Programming Languages

Franklyn Turbak and David Gifford
with Mark A. Sheldon

The MIT Press
Cambridge, Massachusetts
London, England

This book was set in LaTeX by the authors, and was printed and bound in the United States of America.

Library of Congress Cataloging-in-Publication Data

Turbak, Franklyn A.
Design concepts in programming languages / Franklyn A. Turbak and David K. Gifford, with Mark A. Sheldon.

p. cm.

Includes bibliographical references and index.
ISBN 978-0-262-20175-9 (hardcover : alk. paper)

1. Programming languages (Electronic computers). I. Gifford, David K., 1954–. II. Sheldon, Mark A. III. Title.
QA76.7.T845 2008
005.1—dc22 2008013841

10 9 8 7 6 5 4 3 2 1

Preface

This book is the text for 6.821 *Programming Languages*, an entry-level, single-semester, graduate-level course at the Massachusetts Institute of Technology. The students that take our course know how to program and are mathematically inclined, but they typically have not had an introduction to programming language design or its mathematical foundations. We assume a reader with similar preparation, and we include an appendix that completely explains the mathematical metalanguage we use. Many of the exercises are taken directly from our problem sets and examination questions, and have been specifically designed to cause students to apply their newfound knowledge to practical (and sometimes impractical!) extensions to the foundational ideas taught in the course.

Our fundamental goal for *Programming Languages* is to use a simple and concise framework to teach key ideas in programming language design and implementation. We specifically eschewed an approach based on a tour of the great programming languages. Instead, we have adopted a family of syntactically simple pedagogical languages that systematically explore programming language concepts (see Appendix B). Contemporary concerns about safety and security have caused programmers to migrate to languages that embrace many of the key ideas that we explain. Where appropriate, we discuss how the ideas we introduce have been incorporated into contemporary programming languages that are in wide use.

We use an s-expression syntax for programs because this syntactic form is easy to parse and to directly manipulate, key attributes that support our desire to make everything explicit in our descriptions of language semantics and pragmatics. While you may find s-expression syntax unfamiliar at first, it permits the unambiguous and complete articulation of ideas in a simple framework.

Programming languages are a plastic and expressive medium, and we are hopeful that we will communicate our passion for these computational canvases that are an important underpinning for computer science.

Web Supplement

Specialized topics and code that implements many of the algorithms and compilation methods can be found on our accompanying Web site:

`dcpl.mit.edu`

The Web Supplement also includes additional material, such as a section on concurrency and proofs of the theorems stated in the book.

To the Student

The book is full of examples, and a good way to approach the material is to study the examples first. Then review the figures that capture key rules or algorithms. Skip over details that bog you down at first, and return to them later once you have additional context.

Using and implementing novel programming language concepts will further enhance your understanding. The Web Supplement contains interpreters for various pedagogical languages used in the book, and there are many implementation-based exercises that will help forge connections between theory and practice.

To the Teacher

We teach the highlights of the material in this book in 24 lectures over a 14-week period. Each lecture is 1.5 hours long, and students also attend a one-hour recitation every week. With this amount of contact time it is not possible to cover all of the detail in the book. The Web Supplement contains an example lecture schedule, reading assignments, and problem sets. In addition, the MIT OpenCourseWare site at `ocw.mit.edu` contains material from previous versions of 6.821.

This book can be used to teach many different kinds of courses, including an introduction to semantics (Chapters 1–5), essential concepts of programming languages (Chapters 1–13), and types and effects (Chapters 6 and 11–16).

We hope you enjoy teaching this material as much as we have!