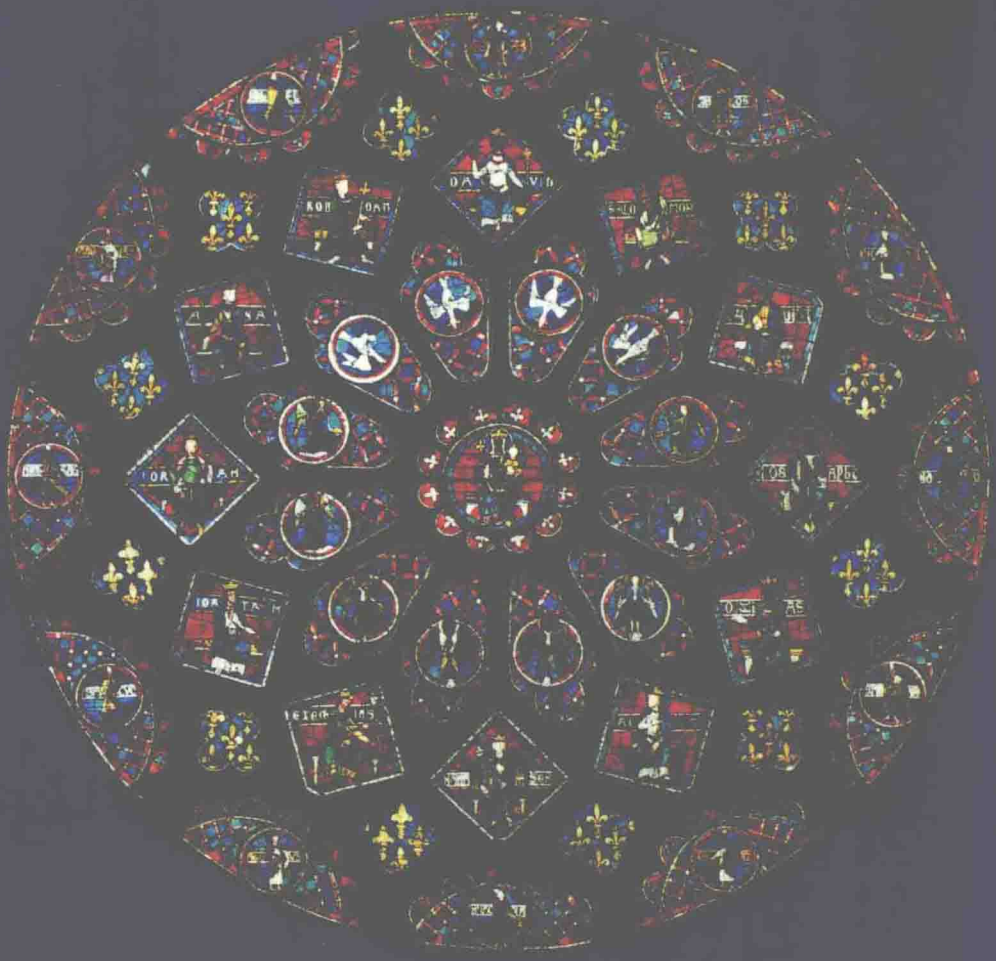
Turbo C++

An Introduction to Computing



JOEL ADAMS
SANFORD LEESTMA
LARRY NYHOFF

Turbo C++: An Introduction to Computing

Joel Adams
Sanford Leestma
Larry Nyhoff

Calvin College
Grand Rapids, Michigan

Prentice Hall, Upper Saddle River, New Jersey 07458

Library of Congress Cataloging-in-Publication Data

Adams, Joel.
Turbo C++: an introduction to computing / Joel Adams, Sanford Leestma, Larry Nyhoff.
p. cm.
Includes index.
ISBN 0-13-439928-5
1. C++ (Computer program language) 2. Turbo C++. I. Leestma, Sanford. II. Nyhoff, Larry R. III. Title.
QA76.73.C153A34 1996
005.13'3—dc20 95-41348
CIP

About the Cover: The rose window on the cover is the Rose de France (c. 1233) in the north transept of the Chartres cathedral. Like many of the other beautiful rose windows in French cathedrals, it is an early example of object-oriented design, in which objects of various shapes, sizes, colors, and meanings are fitted together according to certain basic principles. An interesting property of this window is how its components—the outer semicircles containing the last twelve Old Testament prophets; the quatrefoils containing the three-petaled fleur-de-lis, symbols of the Annunciation and of royalty; the twelve squares containing the kings of the Virgin Mary's ancestry as recorded by St. Matthew; the twelve circles containing doves, angels and other celestial beings; and the twelve-petaled central rosette containing the Virgin Mary—are arranged into spiral-shaped patterns based on the golden section and the Fibonacci sequence.

Acquisitions Editor: Alan Apt
Developmental Editor: Sondra Chavez
Production Manager: Judy Winthrop
Cover Design: Robert Freese
Cover Art: Giraudon/Art Resource LAC 58289. Rose Window, North Transept, Chartres Cathedral (13th C.).
Illustrations: York Graphic Services, Inc.
Manufacturing Buyer: Donna Sullivan

Printed in the United States of America

10 9 8 7 6 5 4 3 2 1

ISBN 0-13-439928-5

PRENTICE-HALL INTERNATIONAL (UK) LIMITED, *London*
PRENTICE-HALL OF AUSTRALIA PTY. LIMITED, *Sydney*
PRENTICE-HALL CANADA, INC., *Toronto*
PRENTICE-HALL HISPANOAMERICANA, S.A., *Mexico*
PRENTICE-HALL OF INDIA PRIVATE LIMITED, *New Delhi*
PRENTICE-HALL OF JAPAN, INC., *Tokyo*
SIMON & SCHUSTER ASIA PTE., LTD., *Singapore*
EDITORA PRENTICE-HALL DO BRAZIL, LTDA., *Rio de Janeiro*

PREFACE

In order to properly introduce students to computing, we believe that the first computer course for students should accomplish two goals:

1. The student should be introduced to the discipline, methodologies, and techniques of computer programming using a modern programming language.
2. The student should be introduced to the breadth of the discipline of computing, so that he or she comes to understand the role of programming in the broader context of computing.

The aim of this textbook is to accomplish both of these goals.

The Programming Goal

In order to accomplish our first goal, we have chosen the language C++, whose features (we believe) make it the language of choice in the immediate future. A few of the reasons for this choice are as follows:

1. C++ provides *a strong type-checking system.*
2. C++ provides *reference parameters* for its functions.
3. C++ provides a *library* mechanism, whereby a programmer can store generally useful functions in a library, so that they can be reused by any program that needs them.
4. C++ provides function name and operator *overloading,* allowing a programmer to use the same name to define subprograms that perform similar operations on different data types.
5. C++ provides *the class,* whereby both an object's data members and its operations can be encapsulated within a single, protected structure.
6. C++ provides *derived classes,* a mechanism whereby one class can *inherit* the data members and operations from another class, allowing a *class hierarchy* to be built.

These latter features of C++ allow it to be characterized as an *object-oriented programming* (OOP) language. From windowing systems to graphic user interfaces to object-oriented databases, more and more of today's best computing appli-

cations are being developed in C++ using the OOP approach, making it today's language of choice.

C++ Is Not C. Many people erroneously believe that C++ is simply its parent language C with a few additional features and that C++ is therefore inappropriate as a first programming language. In fact, most of the design flaws of C have been corrected in C++, making it a suitable language for a first course in computing.

The Importance of Modeling. Another popular misconception is that because many C programs are cryptically written, C++ programs probably suffer the same drawback. We believe that cryptic programs are caused by *people,* not by a *language,* and an undisciplined programmer will write cryptic programs in any language, not just C. This is because:

1. Most people learn C from the examples they see in a book; and
2. Most C books are not introductory programming texts, but rather *language references* intended for professional programmers.

One of the aims of this text is to teach a disciplined programming style (to those with no programming experience) that results in well-documented, easy-to-read programs.

We believe that *what students learn depends on the models that they see.* That is, if they are presented with examples that are well-written, well-documented, and maintainable, then the programs they write will exhibit these same characteristics, regardless of the language used. To that end, this text contains a large number of examples that illustrate good programming style.

Standard C++. At the time of this writing, the American National Standards Institute (ANSI) Committee X3J16 had not completed a C++ standard. In the absence of such a standard, we have used the *Annotated C++ Reference Manual*[1] as our primary reference in preparing this text.

While written in the Turbo environment, the examples in this text are not restricted to that environment and have been successfully ported to the following environments:

Computing Environment	C++ Compiler
UNIX (Sun, Apollo, etc.)	GNU g++ (v. 2.4.5)[2]
DOS, Windows, OS-2 (IBM PC)	Turbo C++ (v. 3.0)[3]
MacOS (Apple Macintosh)	Symantec C++ For Macintosh (v. 6.01)[4]

[1] Margaret Ellis and Bjarne Stroustrup, *The Annotated C++ Reference Manual.* (Addison-Wesley, 1992).

[2] GNU g++ is a copylefted product of the Free Software Foundation, Inc., 675 Mass Ave., Cambridge, MA 02139; and is available for free via anonymous ftp from `prep.ai.mit.edu:/pub/gnu`.

[3] Turbo C++ is a copyrighted product of Borland International Inc., 1800 Green Hills Rd., P.O. Box 660001, Scotts Valley, CA 95067; and is available from most software vendors for the IBM PC.

[4] Symantec C++ for Macintosh is a copyrighted product of Symantec Corporation, 10201 Torre Ave., Cupertino, CA 95014; and is available from most software vendors for the Apple Macintosh.

The Breadth of Computing

In 1991, a new set of curriculum recommendations was published in *Computing Curricula 1991: Report of the ACM/IEEE-CS Joint Curriculum Task Force*. One theme of this report is that an introductory course in computing should introduce the various knowledge areas of the discipline:

- Architecture
- Artificial intelligence and robotics
- Database and information retrieval
- Human–computer communication
- Numerical and symbolic computation
- Operating systems
- Programming languages
- Software methodology and engineering
- Social, ethical, and professional context

In this text, we include a number of sections that illustrate these areas, trying to capture the spirit of these curriculum guidelines in a natural, unobtrusive way. These sections have been carefully selected in accordance with the *Computing Curricula 1991* report to provide an overview of computer science and to provide a foundation for further study in theoretical and/or applied computer science. They have been highlighted in special PART OF THE PICTURE sections, which are marked with an icon in the shape of a puzzle piece. These sections include:

- What Is Computer Science?
- The History of Computing
- Computer Organization
- Social, Professional, and Ethical Issues
- Syntax and Semantics
- Computer Architecture
- Computability Theory
- Introduction to Numeric Computation
- Introduction to Algorithm Analysis
- Simulation
- Artificial Intelligence
- Databases
- Data Encryption
- The Type Hierarchy
- Analysis of Algorithms
- Automata and Language Translation
- Numeric Computation
- Computer Graphics
- Expert Systems

A solid base is thus established for later courses in theoretical and/or applied computer science.

About the Text

Organization. We have organized the text material into four parts:

- The text begins with an *Introduction* consisting of two chapters that present an overview of computing and programming.
- The second part of the text, *Computing with Simple Objects,* consists of five chapters that introduce the student to the basic ideas of computing, including types, variables, constants, functions, I/O, libraries, selective control, repetitive control, parameter passing mechanisms, and so on. Each of these topics is covered in the context of simple data types: the integer, real, character, and boolean types.
- The third part, *Computing with Class Objects,* consists of seven chapters that extend the ideas from the second part to problems involving more sophisticated data types, including files, character strings, enumerations, arrays, and sets. Where applicable, C++ class libraries are used to implement objects that can be easily reused and maintained.
- The final part, *Computing with Advanced Objects,* consists of three chapters that introduce advanced topics, including indirection, run-time allocation/deallocation, and linked structures, such as linked lists, stacks, queues, and trees, each implemented using C++ classes.

We think that most of the first three parts can be covered in a typical semester course. Some or all of the fourth part can be covered in accelerated courses or in a second course or can be used as enrichment material or for honors work.

Features. This first edition text breaks new ground in many ways, by providing a gentle introduction to new topics such as designing for reusability; the use, design, and implementation of class libraries; the overloading of operators and function names; and the OOP approach to program design. A few of the features of the text are:

- Each chapter begins with an example problem, whose solution is used to introduce the ideas of that chapter. Following this example, the concepts and theory behind these ideas are explored, and other examples are presented to reinforce the ideas. In this approach, students see the *practice* of a new topic before the *abstract* definitions and theory behind that topic, providing them with a framework in which those abstract aspects can be organized and understood.
- A wealth of examples illustrates each topic, allowing students to distinguish what is essential from what is optional. In the spirit of *Computing Curricula 1991,* these examples are chosen from a wide range of applications and have been written to model good structure and style. Those marked in the text with a disk icon are included on the data disk that accompanies the Instructor's Manual or can be downloaded from our ftp site as follows:

  ftp to ftp.prenhall.com
  login as anonymous
  use your email address as the password
  cd to pub/EMS/adams/turbo.c++
- Optional sections (marked with asterisks) delve into the more advanced topics, without requiring that they be covered in a normal introductory course.

- Each chapter ends with *Programming Pointers* that highlight important points, especially proper techniques of design and style, as well as common programming pitfalls.
- Color is used to emphasize and highlight important features.
- Exercise sets include short written exercises as well as a large number of programming exercises and projects drawn from a wide range of application areas.

Supplementary Materials

A number of supplementary materials are available from the publisher. These include the following:

- A solutions manual that contains solutions to the exercises in the text, including many of the programming exercises.
- A lab manual and diskette prepared by Professor Joel Adams.
- Disk containing the sample programs and data files referenced in the text. In addition, this material can be downloaded from our ftp site as follows:
  ftp to ftp.prenhall.com
  login as anonymous
  use your email address as the password
  cd to pub/EMS/adams/turbo.c++
- Disks containing all of the text exercises and solutions to many of the programming exercises.

Suggestions

The authors welcome feedback, both positive and negative. Comments about features of the text that work especially well, as well as about features of the text that need improvement, will aid us in the preparation of subsequent editions. We would also appreciate being notified of errors. Such comments can be directed to any of the authors at the following U.S. mail address:

Department of Mathematics and Computer Science
Calvin College
Grand Rapids, Michigan 49546
USA

or to **adams@calvin.edu**, **lees@calvin.edu**, or **nyhl@calvin.edu** via the Internet.

Acknowledgments

The comments and suggestions made by the following reviewers of the forerunner of this text, *C++: An Introduction to Computing,* were valuable, and their work is much appreciated: Jose Cisnaros, Metropolitan College of Denver; Ann Ford, University of Michigan; Mike Holland, Northern Virginia Community College; John Lowther, Michigan Technological University; Dick Reed, Michigan State University; and Peter Spoerri, Fairfield University. Our thanks also go to Evan Scott of

Borland International, Inc., for his work in checking our descriptions of Turbo C++ features in this text. We must also thank our wives, Barbara, Marjory, and Shar, for not complaining about the times that their needs and wants were slighted by our busyness. Above all, we give thanks to God for giving us the opportunity, ability, and stamina to prepare these texts.

J.C.A.
S.C.L.
L.R.N.

CONTENTS