大学计算机教育国外著名教材、教参系列（影印版）

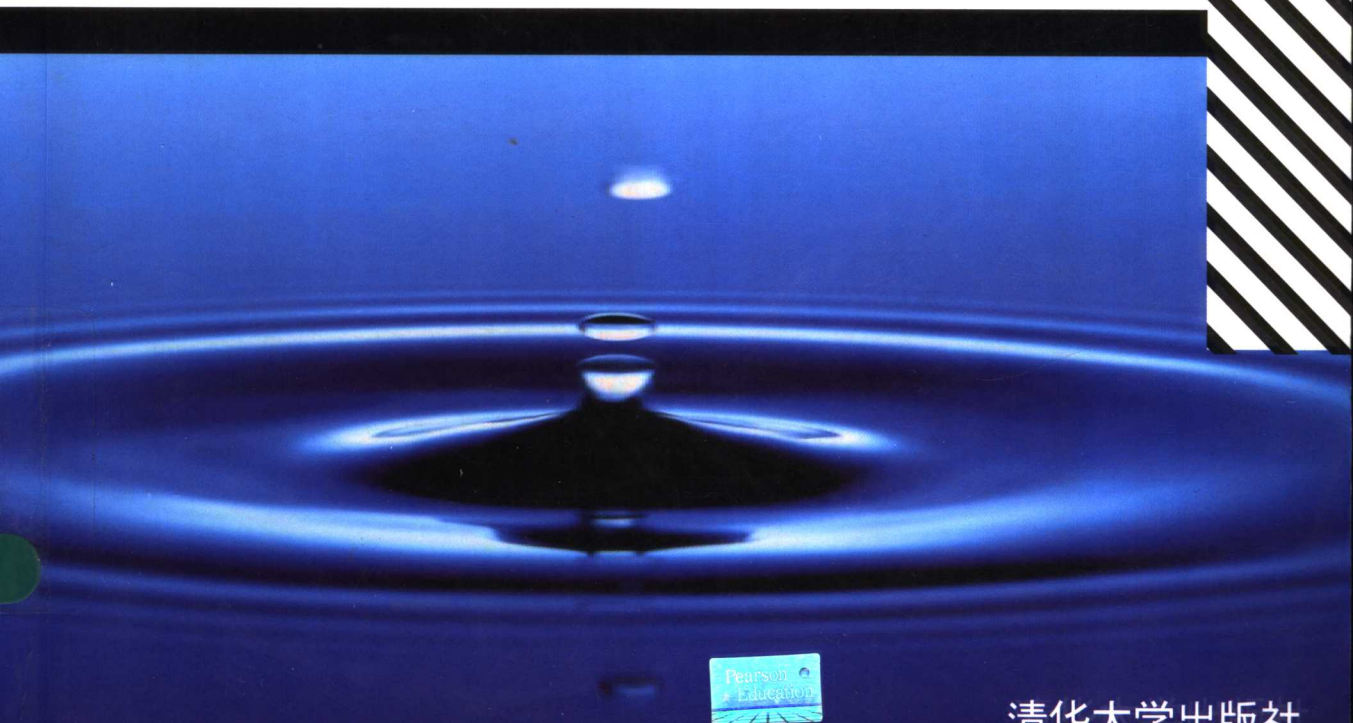
# Data Structures with C++ Using STL

Second Edition

William Ford
William Topp

# 数据结构
# C++语言描述——
## 应用标准模板库（STL）

第2版



清华大学出版社

Data Structures with C++ Using STL

Second Edition

[redacted]

[redacted]

# 数据结构

# C++语言描述

## 使用标准模板库（STL）

大学计算机教育国外著名教材、教参系列（影印版）

# Data Structures with C++ Using STL

Second Edition

# 数据结构 C++语言描述——应用标准模板库（STL）

第 2 版

William Ford

*University of the Pacific*
*Computer Science Department*

William Topp

*University of the Pacific*
*Computer Science Department*

清 华 大 学 出 版 社

EISBN 0-13-085850-1
Data Structures with C++ Using STL Second Edition
William Ford, William Topp


北京市版权局著作权合同登记号：图字：01-2002-3020

**图书在版编目(CIP)数据**

数据结构 C++语言描述. 英文：应用标准模板库（STL）：第 2 版 /（美）福特，（美）托普著. —影印本. —北京：清华大学出版社，2003
（大学计算机教育国外著名教材、教参系列）
ISBN 7-302-06259-5

Ⅰ. 数… Ⅱ. ①福… ②托… Ⅲ. ①数据结构－高等学校－教材－英文 ②算法分析－高等学校－材料－英文 ③C 语言－程序设计－高等学校－教材－英文 Ⅳ. TP311.12

中国版本图书馆 CIP 数据核字（2003）第 004205 号

**出版者**：清华大学出版社（北京清华大学学研大厦，邮编 100084）
http:// www.tup.tsinghua.edu.cn
**印刷者**：北京牛山世兴印刷厂
**发行者**：新华书店总店北京发行所
**开　本**：787×960 1/16　印张：66.5
**版　次**：2003 年 1 月第 1 版　2003 年 1 月第 1 次印刷
**书　号**：ISBN 7-302-06259-5/TP・3745
**印　数**：0001～5000
**定　价**：86.00 元

# Preface

This book is designed to present the fundamentals of data structures from an object-oriented perspective. The focus is on data structures that efficiently store large collections of data. The structures, called containers, feature operations to access, insert, and remove items from the collection. The study of data structures is core to a computer science curriculum. This curriculum has had a rich and storied tradition. Computer researchers and practioners have evolved a wide range of container structures to meet different problem situations. Initially, the focus was on implementation issues so that programs could efficiently store and access large data sets within the limited physical resources of the computer system. As computers developed greater CPU power and increased memory and storage capabilities, researchers and practioners were free to give more consideration to the abstract design of the containers. The efforts were greatly aided by an emerging emphasis on object-oriented programming. Object technology provides a means of viewing containers as objects with designated operations to handle the data. A class declaration defines the structure of a container. The public member functions describe a programming interface that allows a container to be used in applications.

Researchers at AT&T Bell Laboratories and Hewlett-Packard Research Labs combined the principles of generic and object-oriented programming to create a unified approach to the study of data structures and algorithms. The result is the Standard Template Library (STL), which is now part of the standard C++ library. STL provides a modern approach to data structures. It categories the structures as sequence and associative containers, along with adapter classes. By using templates and iterators, the STL library allows a programmer to execute a broad range of algorithms that apply to each of the container classes.

This is, however, not a book on STL. It draws on the design structure of STL to create a unifying study of data structures. The reader will be introduced to the basics

of STL and become familiar with the essential elements of the library. The result will be an appreciation of the power, simplicity, and usefulness of STL. With this background, the reader can easily read a technical book on STL to learn more of its many features. While this book is designed as a textbook, a computer professional could use it as a self-study guide to data structures.

### Approach to Data Structures

This book uses a very careful and systematic approach in the development of each data structure. The reader first views a structure informally as an ADT that provides a description of how the container stores elements. Text, figures, and examples provide a detailed understanding of the key operations for the data structure, without reference to any implementation. The reader is then introduced to a second view of the structure using a formal C++ class declaration or an API. The latter refers to an Application Programming Interface format that is the industry-wide standard for presenting class operations. The API format includes the function prototype, a description of its action, and a listing of its preconditions and postconditions. We use the API format to describe the STL container classes and the class declaration to describe the other data structures that are developed in the book.

Once the reader is familiar with a formal represention of a data structure, the book provides a series of applications, which illustrate problem-solving situations that effectively use the structure. Having the reader understand the implemantion of a data structure is a key feature of the book. Corresponding to each STL container class, the book presents a "miniContainer" class that uses the STL interface but offers a straight forward implementation of the operations. The text clearly presents the design and coding of the key operations. The supplemental software supplies a complete listing of the class, with well-documented code.

### Ins and Outs of the Book

This book assumes the reader has completed a first course in C++ object-oriented programming. The authors assume that the concepts of object composition, operator overloading, pointers and dynamic memory, and inheritance are covered briefly, if at all, in a first course. These concepts are carefully developed in this book in the context of their application to data structures. Periodically the book introduces only the essentials of a programming concept, and makes available a Web Tutorial that develops the concept in greater depth. The tutorials include examples and programs and provide the reader with enrichment that is not critical to understanding the text. The existence of a tutorial is clearly marked with an icon in the margin.

Chapters 1 through 10 cover sequence containers (array, vector, list, deque), the adapter classes (stacks, queues, and priority queues), and an introduction to associative tree containers. The material, along with a development of pointers, dynamic memory, and linked lists contains the topics usually covered in a first course in data structures (CS2).

Chapters 11 through 16 introduce more advanced containers that include sets, maps, balanced trees, heaps, hash tables and graphs. The chapters also include a study of applied searching and sorting algorithms, advanced recursion, and graph algorithms. The material is appropriate for a followup course in advanced data structures and applied algorithms (CS7).

### Supplemental Resources

Readers may access the complete source code listings for all classes and programs in the book from the authors' website at **http://www.uop.edu/fordtopp** or **http://www.fordtopp.com**, and from Prentice Hall at **http://www.prenhall.com**. The C++ source code has been tested and run in the Windows environment using Microsoft Visual C++ and Borland C++ Builder, and in the UNIX environment using GNU C++. The graphics library is implemented in each of these environments.

To successfully compile and run the programs in the book using the Microsoft Visual C++ 6.0 compiler, the reader must install the latest Service Pack. Instructions for obtaining and loading the service pack are available on the authors' and Prentice Hall web sites. The same sites include the Web tutorials and Powerpoint slides that present the key topics from each chapter.

An Instructor's Resource CD (IRCD) is available to instructors and provides answers to all of the written exercises and a solution to all of the programming exercises and programming projects. The IRCD also has sample tests with questions in a variety of formats. All of these elements are provided in Word format (".doc") to enable the selection and modification of individual items. For printing only, the IRCD also supplies the materials in Acrobat Reader (".pdf") and postscript (".ps") format. In addition, it provides individual source files of all the programs (*.cpp) and classes (*.h) that are developed in the exercises. The IRCD is available upon request by Professors and Instructors from your local Prentice Hall sales representative.

### Acknowledgments

The authors have been supported by friends, students, and colleagues throughout the preparation of the second edition of *Data Structures with C++ using STL*. The University of the Pacific has generously provided resources and support to complete the project. Prentice Hall offered a dedicated team of professionals who handled the book design and production. We are especially grateful to our acquisitions editor, Petra Recter and to the production editor, AudriAnna Bazlen. We also appreciate the efforts of Sara Burrows, assistant editor, who worked with us on the compilation of the supplements, and the work of Jennie Burger, who is doing the active marketing of the book.

Students have offered valuable criticism of the manuscript by giving us explicit feedback. Our reviewers offered guidance during the design of the new edition and then followed up with detailed comments on both the content and the pedagog-

ical approach. We took most of their recommendations into account. Thanks go to Carol Roberts, University of Maine; Ken Bosworth, Idaho State University; Ralph Ewton, University of Texas, El Paso. Special thanks go to James Slack at Minnesota State University, Mankato, who made extensive and detailed suggestions. His insights and support were invaluable to the authors and greatly improved the final design and content of the book.

*William Ford*
*William Topp*

# Contents

**Preface** ..... **xxiii**

**1 Introduction to Data Structures** ..... **1**

1-1 WHAT IS THIS BOOK ABOUT? 2

*Data Structures and Algorithms 5*

1-2 ABSTRACT VIEW OF DATA STRUCTURES 5

*The time24 ADT 6*

1-3 AN ADT AS A CLASS 8

*The C++ Class 8*
*Private and Public Sections 9*
*Encapsulation and Information Hiding 9*
*The time24 Class 9*

1-4 IMPLEMENTING C++ CLASSES 13

*Implementation of the time24 Class 14*

1-5 DECLARING AND USING OBJECTS 18

*Running a Program 18*

1-6 IMPLEMENTING A CLASS WITH INLINE CODE 21

*Compiler Use of Inline Code 22*

1-7 APPLICATION PROGRAMMING INTERFACE(API) 23

*Random Numbers 24*
*The randomNumber API 24*
*Application: The Game of Craps 26*

1-8 STRINGS 28

*The string Class 30*
*Additional String Functions and Operations 31*

CHAPTER SUMMARY 36
CLASSES AND LIBRARIES IN THE CHAPTER 37
REVIEW EXERCISES 38

*Answers to Review Exercises 40*

WRITTEN EXERCISES 42
PROGRAMMING EXERCISES 48
PROGRAMMING PROJECTS 51

***2 Object Design Techniques*** ..... **53**

2-1 SOFTWARE DESIGN 55

*Request and Problem Analysis 56*
*Program Design 57*
*Designing the Calendar Class 58*
*Program Implementation 62*
*Implementing the Calendar Class 62*
*Program Testing and Debugging 64*
*Program Maintenance 68*

2-2 HANDLING RUNTIME ERRORS 68

*Terminate Program 69*
*Set a Flag 69*
*C++ Exceptions 70*

2-3 OBJECT COMPOSITION 74

*The timeCard Class 75*
*Implementing the timeCard Class 77*

2-4 OPERATOR OVERLOADING 82

*Operator Functions 85*
*Operator Overloading with Free Functions 86*
*Operator Overloading with Friend Functions 87*
*Overloading Stream I/O Operators 89*
*Member Function Overloading 94*

CHAPTER SUMMARY 97
CLASSES AND LIBRARIES IN THE CHAPTER 98

REVIEW EXERCISES 99

*Answers to Review Exercises 100*

WRITTEN EXERCISES 102
PROGRAMMING EXERCISES 107
PROGRAMMING PROJECTS 108

***3 Introduction to Algorithms*** ..... **113**

3-1 SELECTION SORT 115

*Selection Sort Algorithm 116*

3-2 SIMPLE SEARCH ALGORITHMS 120

*Sequential Search 120*
*Binary Search 122*

3-3 ANALYSIS OF ALGORITHMS 127

*System/Memory Performance Criteria 128*
*Algorithm Performance Criteria: Running Time Analysis 128*
*Big-O Notation 131*
*Common Orders of Magnitude 133*

3-4 ANALYZING THE SEARCH ALGORITHMS 135

*Binary Search Running Time 135*
*Comparing Search Algorithms 136*

3-5 MAKING ALGORITHMS GENERIC 139

*Template Syntax 140*
*Runtime Template Expansion 142*
*Template-Based Searching Functions 144*

3-6 THE CONCEPT OF RECURSION 146

*Implementing Recursive Functions 148*
*How Recursion Works 149*
*Application: Multibase Output 152*

3-7 PROBLEM SOLVING WITH RECURSION 155

*Tower of Hanoi 155*
*Number Theory: The Greatest Common Divisor 159*
*Application of gcd - Rational Numbers 161*
*Evaluating Recursion 164*

CHAPTER SUMMARY 168
CLASSES AND LIBRARIES IN THE CHAPTER 169
REVIEW EXERCISES 169

*Answers to Review Exercises 172*

WRITTEN EXERCISES 173
PROGRAMMING EXERCISES 179
PROGRAMMING PROJECT 182

**4 *The Vector Container*** ..... **183**

4-1 OVERVIEW OF STL CONTAINER CLASSES 184

4-2 TEMPLATE CLASSES 188

*Constructing a Template Class 188*
*Declaring Template Class Objects 191*

4-3 THE VECTOR CLASS 192

*Introducing the Vector Container 195*
*The Vector API 200*

4-4 VECTOR APPLICATIONS 202

*Joining Vectors 203*
*The Insertion Sort 203*

CHAPTER SUMMARY 208
CLASSES AND LIBRARIES IN THE CHAPTER 209
REVIEW EXERCISES 209

*Answers to Review Exercises 211*

WRITTEN EXERCISES 211
PROGRAMMING EXERCISES 216
PROGRAMMING PROJECT 217

**5 *Pointers and Dynamic Memory*** ..... **219**

5-1 C++ POINTERS 221

*Declaring Pointer Variables 222*
*Assigning Values to Pointers 222*
*Accessing Data with Pointers 224*
*Arrays And Pointers 225*
*Pointers and Class Types 227*

5-2 DYNAMIC MEMORY 229

*The Memory Allocation Operator new 229*
*Dynamic Array Allocation 231*
*The Memory Deallocation Operator delete 232*

5-3 CLASSES USING DYNAMIC MEMORY 234

*The Class dynamicClass 234*
*The Destructor 236*

5-4 ASSIGNMENT AND INITIALIZATION *240*

*Assignment Issues 240*
*Overloading the Assignment Operator 242*
*The Pointer this 243*
*Initialization Issues 243*
*Creating a Copy Constructor 244*

5-5 THE MINIVECTOR CLASS 247

*Design of the miniVector Class 248*
*Reserving More Capacity 251*
*The MINIVector Constructor, Destructor, and Assignment 253*
*Adding and Removing Elements from a MINIVector Object 254*
*Overloading the Index Operator 258*

5-6 THE MATRIX CLASS 260

*Describing the Matrix Container 261*
*Implementing Matrix Functions 265*

CHAPTER SUMMARY 266
CLASSES AND LIBRARIES IN THE CHAPTER 267
REVIEW EXERCISES 268

*Answers to Review Exercises 270*

WRITTEN EXERCISES 271
PROGRAMMING EXERCISES 277
PROGRAMMING PROJECT 279

***6 The List Container and Iterators*** ..... **281**

6-1 THE LIST CONTAINER 282

*The list ADT 284*
*The list API 286*
*Application: A List Palindrome 288*

6-2 ITERATORS 290

*The Iterator Concept 290*
*Constant Iterators 294*
*The Sequential Search of a List 296*
*Application: Word Frequencies 298*

6-3 GENERAL LIST INSERT AND ERASE OPERATIONS 302

*Ordered Lists 305*
*Removing Duplicates 307*
*Splicing Two Lists 309*

6-4 CASE STUDY: GRADUATION LISTS 310

*Problem Analysis 310*
*Program Design 310*
*Program Implementation 312*

CHAPTER SUMMARY 315
CLASSES AND LIBRARIES IN THE CHAPTER 316
REVIEW EXERCISES 316

*Answers to Review Exercises 318*

WRITTEN EXERCISES 319
PROGRAMMING EXERCISES 322
PROGRAMMING PROJECT 325

***7 Stacks*** ..... **327**

7-1 THE STACK ADT 328

*Multibase Output 332*
*Uncoupling Stack Elements 336*

7-2 RECURSIVE CODE AND THE RUNTIME STACK 339

7-3 STACK IMPLEMENTATION 342

*miniStack Class Implementation 345*
*Implementation of the STL stack Class (Optional) 346*

7-4 POSTFIX EXPRESSIONS 347

*Postfix Evaluation 349*
*The postfixEval Class 350*

7-5 CASE STUDY: INFIX EXPRESSION EVALUATION 357

*Infix Expression Attributes 358*
*Infix to Postfix Conversion: Algorithm Design 359*
*Infix to Postfix Conversion: Object Design 364*
*infix2Postfix Class Implementation 366*

CHAPTER SUMMARY 372
CLASSES IN THE CHAPTER 373
REVIEW EXERCISES 373

*Answers to Review Exercises 375*

WRITTEN EXERCISES 377
PROGRAMMING EXERCISES 381
PROGRAMMING PROJECTS 382

***8 Queues and Priority Queues*** ..... **384**

8-1 THE QUEUE ADT 386

*Application: Scheduling Queue 388*

8-2 THE RADIX SORT 390

*Radix Sort Algorithm 391*

8-3 IMPLEMENTING THE MINIQUEUE CLASS 395

*Implementation of the STL queue Class (Optional) 398*

8-4 CASE STUDY: TIME-DRIVEN SIMULATION 399

*Simulation Design 400*
*Simulation Implementation 401*

8-5 ARRAY-BASED QUEUE IMPLEMENTATION 406

*Designing the Bounded Queue 409*
*Implementing the Bounded Queue 411*

8-6 PRIORITY QUEUES 412

*Priority Queue ADT 413*
*Sorting with a Priority Queue 415*
*Company Support Services 417*

CHAPTER SUMMARY 421
CLASSES AND LIBRARIES IN THE CHAPTER 422
REVIEW EXERCISES 423

*Answers to Review Exercises 425*

WRITTEN EXERCISES 426
PROGRAMMING EXERCISES 430
PROGRAMMING PROJECT 432

***9 Linked Lists*** ..... **436**

9-1 LINKED LIST NODES 438

*The node Class 439*
*Adding and Removing Nodes 442*

9-2 BUILDING LINKED LISTS 443

*Defining a Singly Linked List 443*
*Inserting at the Front of a Linked List 445*
*Erasing at the Front of a Linked List 447*
*Removing a Target Node 448*

9-3 HANDLING THE BACK OF THE LIST 452

*Designing a New Linked List Structure 453*

9-4 IMPLEMENTING A LINKED QUEUE 455

*The linkedQueue Class 456*
*Implementing the linkedQueue Class 457*

9-5 DOUBLY LINKED LISTS 462

*dnode Objects 463*
*Circular Doubly Linked Lists 466*

9-6 UPDATING A DOUBLY LINKED LIST 468

*The insert() Function 468*
*The erase() Function 470*

9-7 THE JOSEPHUS PROBLEM 474

9-8 THE MINILIST CLASS 477

*miniList Class Private Members 478*
*miniList Class Constructors and Destructor 479*
*Functions Dealing with the Ends of a List 480*
*miniList Iterators 481*
*The miniList Member Functions begin() and end() 485*
*The General List Insert Function 485*

9-9 SELECTING A SEQUENCE CONTAINER 486

CHAPTER SUMMARY 487
CLASSES AND LIBRARIES IN THE CHAPTER 489
REVIEW EXERCISES 489

*Answers to Review Exercises 493*

WRITTEN EXERCISES 495
PROGRAMMING EXERCISES 498
PROGRAMMING PROJECT 500

**10 Binary Trees ..... 502**

10-1 TREE STRUCTURES 504

*Tree Terminology 505*
*Binary Trees 506*

10-2 BINARY TREE NODES 510

*Building a Binary Tree 511*

10-3 BINARY TREE SCAN ALGORITHMS 514

*Recursive Tree Traversals 514*
*Iterative Level-Order Scan 518*

10-4 USING TREE SCAN ALGORITHMS 522

*Computing the Leaf Count 522*
*Computing the Depths of a Tree 523*
*Copying a Binary Tree. 526*
*Deleting Tree Nodes. 529*
*Displaying a Binary Tree. 530*

10-5 BINARY SEARCH TREES 532

*Introducing Binary Search Trees 533*
*Building a Binary Search Tree 534*
*Locating Data in a Binary Search Tree 535*
*Removing a Binary Search Tree Node 536*
*A Binary Search Tree Class 537*
*Access and Update Operations 538*

10-6 USING BINARY SEARCH TREES 543

*Application: Removing Duplicates 543*
*Application: The Video Store 545*

10-7 IMPLEMENTING THE stree CLASS 551

*The stree Class Data Members 553*
*Constructor, Destructor, and Assignment 554*
*Update Operations 554*
*Complexity of Binary Search Tree Operations 563*

10-8 THE STREE ITERATOR (Optional) 563

*Implementing the stree Iterator 565*

CHAPTER SUMMARY 569
CLASSES AND LIBRARIES IN THE CHAPTER 571
REVIEW EXERCISES 571

*Answers to Review Exercises 574*

WRITTEN EXERCISES 576
PROGRAMMING EXERCISES 579
PROGRAMMING PROJECTS 581

***11 Associative Containers*** ..... **586**

11-1 OVERVIEW OF ASSOCIATIVE CONTAINERS 587

*Associative Container Categories 587*
*STL Associative Containers 590*
*Implementing Associative Containers 590*

11-2 SETS 591

*Displaying a Container Using Iterators 592*
*Set Access and Update Functions 593*
*A Simple Spelling Checker 596*
*Application: Sieve of Eratosthenes 600*
*Set Operations 603*
*Application: Updating Computer Accounts 606*

11-3 MAPS 610

*The Map Class Interface 610*
*Map Operations 613*