Data Structures and Algorithms Using C#

C# programmers: no more translating data structures from C++ or Java to use in your programs! Mike McMillan provides a tutorial on how to use data structures and algorithms plus the first comprehensive reference for C# implementation of data structures and algorithms found in the .NET Framework library, as well as those developed by the programmer.

The approach is very practical, using timing tests rather than Big O notation to analyze the efficiency of an approach. Coverage includes array and ArrayLists, linked lists, hash tables, dictionaries, trees, graphs, and sorting and searching algorithms, as well as more advanced algorithms such as probabilistic algorithms and dynamic programming. This is the perfect resource for C# professionals and students alike.

Michael McMillan is Instructor of Computer Information Systems at Pulaski Technical College, as well as an adjunct instructor at the University of Arkansas at Little Rock and the University of Central Arkansas. Mike's previous books include *Object-Oriented Programming with Visual Basic.NET, Data Structures and Algorithms Using Visual Basic.NET*, and *Perl from the Ground Up*. He is a co-author of *Programming and Problem-Solving with Visual Basic.NET*. Mike has written more than twenty-five trade journal articles on programming and has more than twenty years of experience programming for industry and education.

Data Structures and Algorithms Using C#

Michael McMillan
Pulaski Technical College

CAMBRIDGE
UNIVERSITY PRESS

CAMBRIDGE UNIVERSITY PRESS
Cambridge, New York, Melbourne, Madrid, Cape Town, Singapore, São Paulo

Cambridge University Press
32 Avenue of the Americas, New York, NY 10013-2473, USA

www.cambridge.org
Information on this title: www.cambridge.org/9780521670159

First published 2007

Printed in the United States of America

A catalog record for this publication is available from the British Library.

Library of Congress Cataloging in Publication Data

McMillan, Michael, 1957–
Data structures and algorithms using C# / Michael McMillan.
p. cm.
Includes bibliographical references and index.
ISBN-13: 978-0-521-67015-9 (pbk.)
ISBN-10: 0-521-67015-2 (pbk.)
1. C# (Computer program language) I. Title. II. Series.
QA76.73.C154M43 2006
005.13′3–dc22 2006024382

ISBN 978-0-521-87691-9 hardback
ISBN 978-0-521-67015-9 paperback

Cambridge University Press has no responsibility for
the persistence or accuracy of URLs for external or
third-party Internet Web sites referred to in this publication
and does not guarantee that any content on such
Web sites is, or will remain, accurate or appropriate.

Cambridge University Press
978-0-521-87691-9 - Data Structures and Algorithms Using C#
Michael McMillan
Frontmatter
More information

Contents

Preface

The study of data structures and algorithms is critical to the development of the professional programmer. There are many, many books written on data structures and algorithms, but these books are usually written as college textbooks and are written using the programming languages typically taught in college—Java or C++. C# is becoming a very popular language and this book provides the C# programmer with the opportunity to study fundamental data structures and algorithms.

C# exists in a very rich development environment called the .NET Framework. Included in the .NET Framework library is a set of data structure classes (also called collection classes), which range from the Array, ArrayList, and Collection classes to the Stack and Queue classes and to the HashTable and the SortedList classes. The data structures and algorithms student can now see how to use a data structure before learning how to implement it. Previously, an instructor had to discuss the concept of, say, a stack, abstractly until the complete data structure was constructed. Instructors can now show students how to use a stack to perform some computation, such as number base conversions, demonstrating the utility of the data structure immediately. With this background, the student can then go back and learn the fundamentals of the data structure (or algorithm) and even build their own implementation.

This book is written primarily as a practical overview of the data structures and algorithms all serious computer programmers need to know and understand. Given this, there is no formal analysis of the data structures and algorithms covered in the book. Hence, there is not a single mathematical formula and not one mention of Big Oh analysis (if you don't know what this means, look at any of the books mentioned in the bibliography). Instead, the various data structures and algorithms are presented as problem-solving tools.

Simple timing tests are used to compare the performance of the data structures and algorithms discussed in the book.

PREREQUISITES

The only prerequisite for this book is that the reader have some familiarity with the C# language in general, and object-oriented programming in C# in particular.

CHAPTER-BY-CHAPTER ORGANIZATION

Chapter 1 introduces the reader to the concept of the data structure as a collection of data. The concepts of linear and nonlinear collections are introduced. The Collection class is demonstrated. This chapter also introduces the concept of generic programming, which allows the programmer to write one class, or one method, and have it work for a multitude of data types. Generic programming is an important new addition to C# (available in C# 2.0 and beyond), so much so that there is a special library of generic data structures found in the System.Collections.Generic namespace. When a data structure has a generic implementation found in this library, its use is discussed. The chapter ends with an introduction to methods of measuring the performance of the data structures and algorithms discussed in the book.

Chapter 2 provides a review of how arrays are constructed, along with demonstrating the features of the Array class. The Array class encapsulates many of the functions associated with arrays (UBound, LBound, and so on) into a single package. ArrayLists are special types of arrays that provide dynamic resizing capabilities.

Chapter 3 is an introduction to the basic sorting algorithms, such as the bubble sort and the insertion sort, and Chapter 4 examines the most fundamental algorithms for searching memory, the sequential and binary searches.

Two classic data structures are examined in Chapter 5: the stack and the queue. The emphasis in this chapter is on the practical use of these data structures in solving everyday problems in data processing. Chapter 6 covers the BitArray class, which can be used to efficiently represent a large number of integer values, such as test scores.

Strings are not usually covered in a data structures book, but Chapter 7 covers strings, the String class, and the StringBuilder class. Because so much data processing in C# is performed on strings, the reader should be exposed to the special techniques found in the two classes. Chapter 8 examines the use of regular expressions for text processing and pattern matching. Regular expressions often provide more power and efficiency than can be had with more traditional string functions and methods.

Chapter 9 introduces the reader to the use of dictionaries as data structures. Dictionaries, and the different data structures based on them, store data as key/value pairs. This chapter shows the reader how to create his or her own classes based on the DictionaryBase class, which is an abstract class. Chapter 10 covers hash tables and the HashTable class, which is a special type of dictionary that uses a hashing algorithm for storing data internally.

Another classic data structure, the linked list, is covered in Chapter 11. Linked lists are not as important a data structure in C# as they are in a pointer-based language such as C++, but they still have a role in C# programming. Chapter 12 introduces the reader to yet another classic data structure—the binary tree. A specialized type of binary tree, the binary search tree, is the primary topic of the chapter. Other types of binary trees are covered in Chapter 15.

Chapter 13 shows the reader how to store data in sets, which can be useful in situations in which only unique data values can be stored in the data structure. Chapter 14 covers more advanced sorting algorithms, including the popular and efficient QuickSort, which is the basis for most of the sorting procedures implemented in the .NET Framework library. Chapter 15 looks at three data structures that prove useful for searching when a binary search tree is not called for: the AVL tree, the red-black tree, and the skip list.

Chapter 16 discusses graphs and graph algorithms. Graphs are useful for representing many different types of data, especially networks. Finally, Chapter 17 introduces the reader to what algorithm design techniques really are: dynamic algorithms and greedy algorithms.

ACKNOWLEDGEMENTS

There are several different groups of people who must be thanked for helping me finish this book. First, thanks to a certain group of students who first sat through my lectures on developing data structures and algorithms. These students include (not in any particular order): Matt Hoffman, Ken Chen, Ken Cates, Jeff Richmond, and Gordon Caffey. Also, one of my fellow instructors at Pulaski Technical College, Clayton Ruff, sat through many of the lectures and provided excellent comments and criticism. I also have to thank my department dean, David Durr, and my department chair, Bernica Tackett, for supporting my writing endeavors. I also need to thank my family for putting up with me while I was preoccupied with research and writing. Finally, many thanks to my editors at Cambridge, Lauren Cowles and Heather Bergman, for putting up with my many questions, topic changes, and habitual lateness.