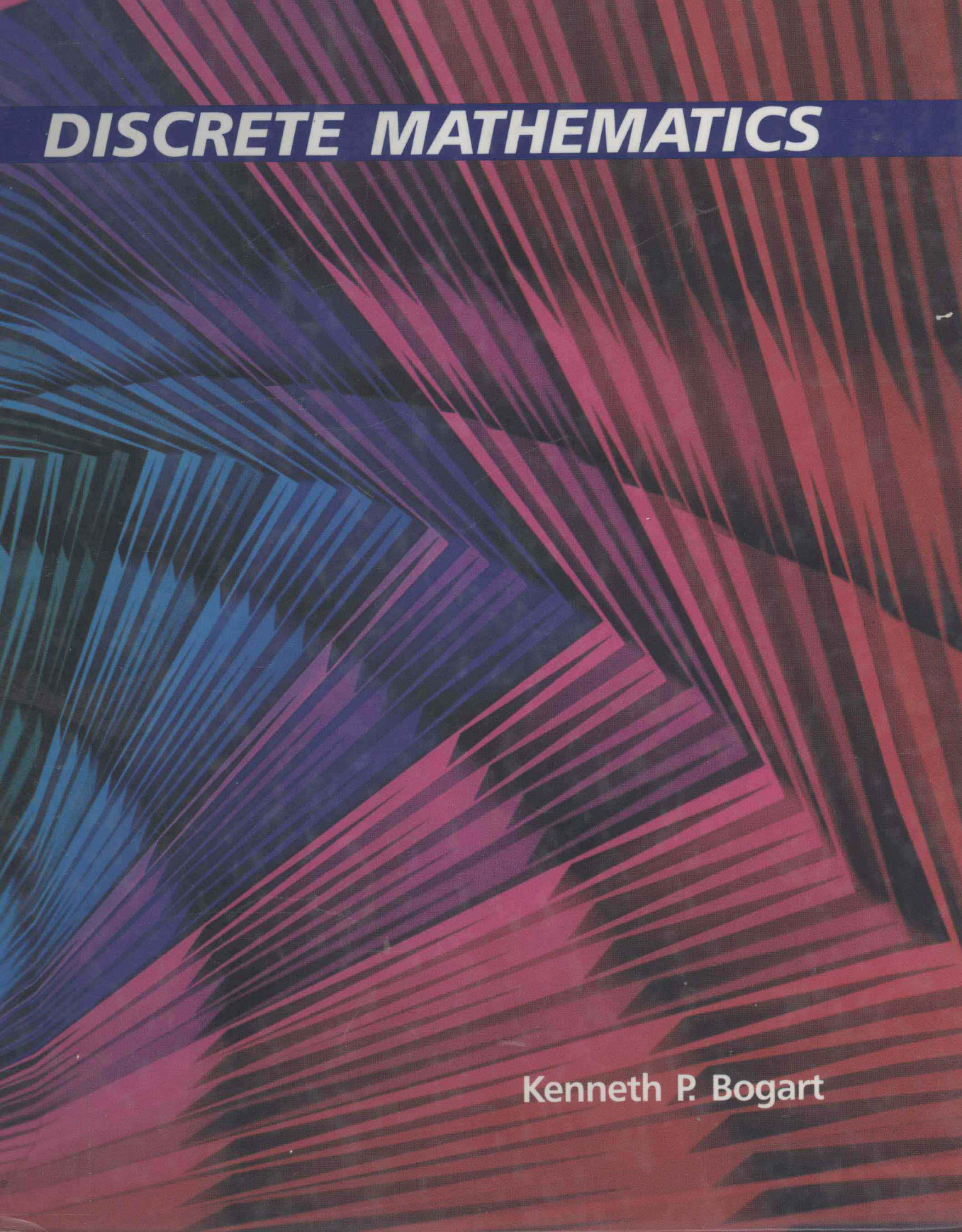
DISCRETE MATHEMATICS



Kenneth P. Bogart

DISCRETE MATHEMATICS

Kenneth P. Bogart
Dartmouth College

D. C. HEATH AND COMPANY
Lexington, Massachusetts Toronto

Acquisitions Editor: Mary Lu Walsh

Developmental Editor: Ann Marie Jones

Production Editor: Karen Potischman

Designer: Cornelia Boynton

Production Coordinator: Mike O'Dea

Cover: Henry Reis

Published simultaneously in Canada.

Printed in the United States of America.

International Standard Book Number: 0-669-08665-7

Library of Congress Catalog Card Number: 87-80567

5 6 7 8 9 0

Preface

There is a growing awareness in the mathematical community that the mathematical needs of students in their first two years of college include a number of subjects not touched on in calculus and touched on only lightly in a traditional high school curriculum. These topics—combinatorics, logic, relations and functions, mathematical induction, graphs and trees, probability, linear algebra, and other modern algebra topics—have in common a step-by-step, or discrete, nature (rather than a continuous nature) and so are referred to as *discrete mathematics*. While these subjects have important applications in the physical, engineering, management, and social sciences, it has been possible to delay their introduction or to treat them in an ad hoc manner as they arise in other courses. However, computer scientists have found that students need to know topics in discrete mathematics in order to fully understand even introductory computer science. The importance of computer science (and therefore discrete mathematics) to computing in other disciplines is becoming more apparent as computers are used in more and more complex situations. Thus discrete mathematics is quickly becoming an essential early ingredient in the college education of many students.

Purpose

After analyzing student needs, a panel of the Mathematical Association of America has issued recommendations for a two-semester course in discrete mathematics. The work of this panel is based first on the panel members' considerable expertise (including Alfs Berztiss's and Anthony Ralston's insights published in the *Communications of the ACM* and the *American Mathematical Monthly*), second on prior recommendations of the ACM curriculum committee and IEEE educational activities board and simultaneous recommendations of the ACM task force on computing programs in small colleges, and finally on many other sources including a D. C. Heath survey of approximately 1500 departments of mathematics and computer science. D. C. Heath's survey indicates that present discrete mathematics courses at the elementary level are largely one semester. Recognizing this possibility, the panel has also recommended a syllabus for a one-semester course. This book has been based on the panel recommendations and also addresses the needs outlined in the other studies. The book is designed to be used in a one- or two-semester course and is constructed to allow the instructor of a one-semester course to choose topics according to institutional needs. While the instructor can enrich the course for students who have already had calculus, calculus is not used in any

of the exposition and algebra topics traditionally used in calculus are reviewed as they arise. Thus the book is suited to students who have not had calculus and will help prepare them for calculus.

Because computer science or computer use in other fields is important to almost all students taking the course, applications of discrete mathematics to computing are used for motivation throughout the book. Appropriate examples from areas outside computer science help indicate the breadth of applications of discrete mathematics. A central theme in the book is that in order to solve problems by computer, one needs algorithms, and in order to evaluate algorithms, one needs to predict their running time.

Organization

In order to maximize the flexibility of the book, it has been written so that the last section (or sections) in each chapter (other than Chapter 4 on functions) may be omitted without seriously curtailing what can be taught later. Each chapter of the book is divided into sections, which are divided further into subsections (usually two) of approximately equal length and difficulty. The exercises are keyed to subsections, making it easy to adjust the pace of the course to one, two, or three subsections a day, depending on the students' background and the emphasis of the course.

The following table shows the immediate prerequisites for various sections of the book. Of course a prerequisite of a prerequisite is a prerequisite. Except as shown, earlier sections of a chapter are prerequisite for later sections. A section is marked as enriching but not prerequisite if it either provides background for a deeper understanding of the concepts introduced or provides the background needed for an isolated example in that section.

Sections	*Immediate Prerequisites*	*Enriching, but not Prerequisite*
1-3, 1-4	1-1, 1-2	
1-5	1-3, 1-4	
2-1, 2-2, 2-3	1-3, 1-4	1-5
3-1, 3-2, 3-3A	1-1, 1-2	
3-3B	3-1, 3-2, 3-3A	
3-4	3-1, 3-2	3-3A
4-1, 4-2, 4-3	3-1, 3-2, 3-3A	
5-1, 5-2	4-1, 4-2, 4-3	2-1, 2-2, 2-3
5-3A	5-1, 5-2	
5-3B	5-3A	
6-1	5-2	
6-2	6-1	
6-3	6-2	
6-4	6-2	
6-5	6-2, 6-4	6-3
7-1, 7-2	6-2, 6-4	5-3A, 6-3, 6-5

Sections	*Immediate Prerequisites*	*Enriching, but not Prerequisite*
7-3	7-1, 7-2A	
8-1	6-2, 6-4	
8-2A	8-1	
8-2B, 8-3	5-3A, 8-2A, 7-2	
9-1, 9-2, 9-3	8-1	6-4
9-4	9-1, 9-2, 9-3	5-3B
10-1	5-1, 4-2	
10-2	10-1, 6-1	8-4, 9-1, 9-2, 9-3
10-3	10-1	
10-4, 10-5, 10- 6	10-3	
11-1, 11-2	6-2, 6-4	6-3
11-3	11-1, 11-2	7-1, 7-2, 8-1, 8-2A
11-4	11-3	
11-5	11-3, 10-1	10-2
12	10-3, 1-5	9-4, 10-4, 10-5, 10-6, 2-1, 2-2, 2-3

Students with a strong algebra background (such as a thorough preparation for calculus) may find the material on sets, functions, and relations to be largely review. Instructors whose students have such a background may wish to substitute Appendix A, which is a condensed review of these topics from a discrete mathematics point of view, for Sections 1-1 and 1-2, the first half (except for adjacency lists) of Section 3-1, and Sections 4-1 and 4-2.

Contents

Chapter 1 begins with an introduction to sets as truth sets of statements. Later, optional, material in Chapter 1 covers truth tables for the basic connectives and their applications. I have adopted the method of working out truth tables popularized by Kemeny, Snell, et al. in their *Finite Mathematics* series. This method avoids the need to define and analyze subformulas and, in complicated situations, saves the students a good bit of writing. The subject takes on special importance in light of the use of compound statements to control conditional instructions in computer programs. While the topics in Chapter 1 are usually thought of as propositional logic, the applications in programming require the use of variables. For this reason the typical introductory study of propositions is replaced by a parallel study of statements about a universe. Chapter 2 continues the study of logic through rules of inference and the use of quantifiers. The chapter allows the instructor who wishes to concentrate on the nature of proof to spend considerable time there while allowing the instructor whose goals are to help students learn to read proofs and recognize quantifications to proceed more quickly. The treatment of logic is designed to prepare students for later work in the theory of computation, databases, and artificial intelligence. Instructors may find that their students already have an understanding of proof techniques that is appropriate for the goals of their course. In this case it is possible to omit the second half of Chapter 1 and/or Chapter 2 with little or no effect on what can be covered later.

Chapters 3 and 4 are an introduction to relations and functions. Directed graphs (digraphs) and graphs are introduced as tools for the study of relations and symmetric relations, respectively. As there is no standard accepted definition of *digraph* or *graph* in the mathematical community, I've adopted the definitions which fit best with the relational point of view. Equivalence relations are introduced in Chapter 3 and they arise frequently in later chapters; congruence modulo m and partial orderings are also introduced in Chapter 3, but they may be downplayed or omitted as their later use is minor. Instructors who plan to cover Chapter 12 (Abstract Algebra) will want to cover congruence modulo m at some time, however. All of Chapter 4 is fundamental to the remainder of the book. Functions are the basis for the study of permutations and other topics in combinatorics and relative growth rates of functions play an essential role in later algorithm analysis. Students who have had a solid precalculus course can omit or cover quickly all but the last section of Chapter 4; conversely all of Chapter 4 is a good preparation for calculus. The last section introduces the "big Oh" notation which is so useful in discussing algorithms. The functions discussed actually arise in analyzing algorithms, so this section is central to what comes later.

Chapter 5 is a study of the principle of mathematical induction. By Chapter 5 the student has enough background to understand a wide variety of applications. In addition to the sum formulas the student may have seen in algebra, the examples include results about graphs, proving the correctness of algorithms, proving statements about how many steps an algorithm uses, and even specifying algorithms themselves. We also discuss how induction lies at the heart of grammars, which define parts of computer languages. In this context, induction is a core topic in computer science as well as in mathematics! While the study of Chapter 5 is enriched by a study of Chapter 2, Chapter 5 relies only on Sections 1-1 and 1-2, and on Chapters 3 and 4.

Chapter 6 gives the student the basic combinatorial tools used in all counting problems, including counting the number of steps used by an algorithm. We begin by applying the sum and product principles to permutations and combinations, viewed as one-to-one functions and subsets. We move on to discuss a number of applications of basic counting principles, including the binomial theorem. The chapter then unifies a large number of standard applications of the product principle in terms of counting equivalence classes, and introduces multisets (combinations with repetitions). The instructor may choose the brief treatment of multisets at the end of Section 6-2, or the more thorough treatment in Section 6-3 (or both). Chapter 6 concludes with a complete (and optional) discussion of the principle of inclusion and exclusion.

The remaining chapters are largely independent of each other and can be covered as local needs dictate. Chapter 7 covers recurrence relations as a means of unifying and extending previous approaches to solving counting problems. The discussion of first-order recurrence relations and divide-and-conquer algorithms contains important computer science examples (such as binary search and sorting algorithms) and gives the student a new perspective on the importance of logarithms and exponential functions. The treatment of second-order recurrence rela-

tions is based on a study of growth problems that includes the Fibonacci numbers in a natural way. Appendix B, on generating functions, is an extension of Chapters 6 and 7. This material unifies many topics in enumeration and is suitable for students with a strong background in algebra. It is especially appropriate for students who have taken or are about to take a course in the calculus of power series.

Chapters 8 and 9 give a solid introduction to graph theory. Chapter 8 begins with trees defined in graph theory terms and quickly progresses to the rooted and binary trees used in computer science. The optional material on two-three trees answers a practical question students may well ask: "What good are binary trees if our data is already (almost) in order?" These two-three trees are special cases of B-trees, a data structure that lies at the heart of modern database management systems. The final section of Chapter 8 covers optimal spanning trees, giving procedures for finding minimum cost communications networks and minimum length paths in graphs.

Chapter 9 introduces a cross-section of other topics in graph theory, and relies on Section 8-1. The instructor who wishes to give a brief overview of graphs may do so by choosing to downplay the role of proofs. On the other hand, for the instructor whose goals include teaching students to write clear and concise proofs, this is an ideal chapter in which to slow down and spend time working on proof technique. The chapter begins with a discussion of various graphical structures and isomorphism. We distinguish between the problem of verifying that a given function is an isomorphism and the problem of determining whether two graphs are isomorphic. This distinction lies at the heart of the definition of the complexity class NP; while it may be unrealistic to try to define and use NP in this course, we present topics in a way that will make them as useful as possible if the student later studies complexity theory. The surprising observation that while Eulerian tours are relatively straightforward to discover, little is known about Hamiltonian tours again motivates a study of complexity theory in later courses. Planarity is now a practical subject since the chips of silicon on which we lay our circuits are planar! Colorability too has its practical aspects in scheduling. We mention briefly how these two subjects combine in the four-color theorem. Directed graphs provide an excellent way to study machines; we introduce finite state machines and automata and study their relationships with the grammars and languages introduced in Chapter 5.

Matrix algebra has applications throughout discrete mathematics. Also, the later study of linear algebra is enhanced by an intuitive understanding of matrix algebra. For these reasons we devote Chapter 10 to matrix algebra. After a discussion of the basic matrix operations we study applications of matrices to graphs, including transitive closures and distances in graphs. (This chapter can be taken up anytime after Section 6-1.) Discussing the solution of systems of equations as matrix equations leads to the inverse of a matrix. Our final topic is determinants, which we approach as numbers used to test the invertibility of matrices. The chapter gives an elementary treatment via row operations.

The subject of probability has many applications throughout the sciences and engineering. Among the reasons it is important in computer science is the concept

of expected value. The second most practical question about a program is "How long does it take?" This leads to the concept of the expected value of the running time. While we discuss many applications in the examples, exercises, and problems of Chapter 11, it is possible to illustrate most ideas of probability using simple examples (coin tossing, test taking, and so on). In order to keep the treatment of probability as elementary as possible, we rely on these basic examples for motivation and introduction of new ideas. The discussion of expected value includes a discussion of the expected running time of algorithms and the depth of binary search trees, and culminates in the central limit theorem. Here we illustrate for students the meaning of such common (and misunderstood) phrases as "95% sure." How, for example, can we be 95% sure (or 99.9% sure) that a database will not overflow its alloted space? We conclude our study with another application of matrix algebra, Markov chains.

Abstract algebra and number systems are handled differently in this text than in others. I have found that students regard an unmotivated study of number systems and their properties as an unnecessary rehash of elementary school mathematics, so I prefer to briefly review these topics as they become relevant throughout the text. Definitions of algebraic systems such as groups, semigroups, rings, monoids, and so on, have been saved for the last chapter. The topics chosen are important both in computer science and mainstream mathematics. The treatment begins by identifying abstractions of properties rather than systems. It then introduces the algebraic systems as examples having these properties, and develops the theory of the systems themselves. The properties abstracted are taken up over and over in early chapters of the book for the instructor who chooses to emphasize them, allowing the algebra to appear as part of a natural progression (as, of course, it is). This early exposure includes the treatment of concrete examples of Boolean algebras in Chapter 1, congruence classes in Chapter 3, inductive proofs of facts of arithmetic in Chapter 5, constructions of new machines from old in the problems of Section 9-4, and especially, matrix algebra in Chapter 10. There is a complete but elementary discussion of what we usually call "the first fundamental homomorphism theorem." The concepts of groups, rings, fields, matrices, and cosets are all reinforced by the treatment of error correcting codes in the last section. This chapter is designed to prepare the student for abstract thinking and is ideal preparation for an abstract algebra course.

A few topics that appear often in upper-level discrete mathematics courses have been downplayed or omitted in this book. For example, Karnaugh maps and the Quine-McCluskey procedure for minimizing the number of "logic gates" in a circuit are less important now than they have been in the past. In designing VLSI chips it is not important to minimize logic gates since they occur when "wires" cross in different layers; other design criteria such as heat and deviations from planarity are more important. This is an example in which concepts relevant to one technology are less relevant to another. Minimization of logic gates will be an important topic in digital circuit theory at least until VLSI compilers and chip fabrication technologies become far more available and cheaper. However, this topic is best saved for a specialized course in digital circuits rather than a general course in discrete mathematics.

Exercises and Problems

The **Exercises** have been designed so that each major example or concept has at least two exercises, one odd-numbered and one even-numbered, to go with it. The exercises are intended to give the student mental exercise, and while we hope to make exercises (whether physical or intellectual) interesting and sometimes challenging, we hope above all to make them things that people can and will do. Most exercises give the student one straightforward operation to carry out, as do exercises in courses such as algebra or calculus. Many exercises allow the student to emulate an example or a discussion in the text, leading to further study of the text. However, the process the student finds to emulate is often a thought process, and thus by doing the exercise the student begins to learn to think independently.

For the student (or instructor) not sufficiently challenged by the exercises there is a set of **Problems** at the end of each section. Problems are intended to be interesting, to extend the ideas in the text, to be challenging, and by sometimes including open-ended questions that the student must interpret before answering, to provide a ground for experimentation with generating new questions. Problems allow an instructor to teach a more sophisticated and challenging course.

At the end of each section there is also a fill-in-the-blanks **Concepts Review** with leading questions that will help the student recall major points or definitions from the section just covered. These are helpful for "pulling together" the ideas of a section before attempting the exercises, and also for later review.

The **Review Exercises** at the end of each chapter are exercises in the same sense described above. Some bring together ideas from more than one section of a chapter or ask a question in a different way; most, however, are intended to review what the student has learned in doing earlier exercises.

Solutions to odd-numbered section exercises and to all concepts reviews and chapter review exercises appear in the back of the book, following a list of suggestions for further reading keyed to individual chapters and sections.

Supplements

The **Instructor's Guide** contains solutions to all the even-numbered exercises, discussion of how to do all the problems, possible schedules, and advice and teaching aids for instructors who are relatively new to teaching discrete mathematics.

The book **Introductory Programming for Discrete Mathematics** contains supplementary material on computer programming that illustrates how the ideas of this course are applied in practice. It is intended to be an introduction to computing for students taking a course in discrete mathematics. It can be integrated into a course; however it is also designed for use as a "laboratory manual." While the programs in the book should run in any ANSI standard BASIC that supports recursion and structured programming, they have all been written and tested in True BASIC™. I have collected these programs onto a disk that is ready to run with the user's True BASIC™ package and is available on request from D. C. Heath for the IBM PC™, PC II™, the Apple Macintosh™, the Commodore Amiga™, and the Atari SE™ computers.

The True BASIC™ **Discrete Mathematics software package,** designed and written by John Kemeny, is also available from D. C. Heath. This is a stand-alone program students may use to illustrate various aspects of the course without doing any programming or developing any special computer skills. This program is also ideal for classroom demonstrations of various discrete mathematics topics.

Acknowledgements

This book has been tried out by a wide variety of Dartmouth students and faculty members (ranging from graduate students to full professors); in addition, portions were used at Northeastern University and portions were used in an intensive summer seminar for high school teachers. To all these students and instructors, I offer my thanks for their patience and advice. In addition, a wide variety of people from the mathematics and computer science community have read and commented on portions of the text. They include: David Berman, University of New Orleans; Douglas Campbell, Brigham Young University; Margaret B. Cozzens, Northeastern University; W. E. Deskins, University of Pittsburgh; Thomas A. Dowling, Ohio State University; Andrew G. Earnest, Southern Illinois University; Lyenatte S. Goff, Glendale Community College; Keith Harrow, Brooklyn College, CUNY; Stuart H. Hirshfield, Hamilton College; John Kenelly, Clemson University; Joseph Malkevitch, York College, CUNY; Thomas J. Myers, University of Delaware; John D. Neff, Georgia Institute of Technology; Richard K. Oliver, Indiana State University; Richard Palais, Brandeis University; Fred Roberts, Rutgers University; and Henry M. Walker, Grinnel College. Ann Marie Jones at D. C. Heath played an essential role in helping me integrate this feedback into meaningful change of my original manuscript. Her unique perspective as both an editor and former high school mathematics teacher resulted in better explanations and made my expectations of students far more realistic! Karen Potischman has blended her considerable skills as a production editor with a sympathy for an author's concerns that have made the production process a pleasure. Barry Willenbring has checked my solutions to the odd-numbered and review exercises. To everyone who has helped shape this book, I offer my thanks.

Contents

CHAPTER 1
Sets and Statements

In this chapter, we learn about sets and statements and how they relate to each other. The truth set of a statement, introduced in this chapter, arises again when we study probability, where it can help us find a numerical measure of how likely the statement is to be true. The concept of truth or falsity of a statement will be an important part of our study of logic in Chapter 2. Throughout the remainder of the book, the concept of a set will be a fundamental building block for the rest of our work, especially our study of relations in Chapter 3, functions in Chapter 4, trees and graphs in Chapters 8 and 9, probability in Chapter 11, and algebra in Chapter 12. It is possible to give a development of the theory of sets analogous to Euclid's development of plane geometry. This approach to set theory can occupy an entire course in mathematics, so instead we rely on our intuition as the foundation of our study of sets.

In this chapter, we introduce the concept of an algorithm—the name for a list of instructions for carrying out a process. This concept will arise in different ways throughout our study. At times we shall develop algorithms to solve certain kinds of problems, and at other times we shall develop tools to understand our algorithms better. We shall see how the truth or falsity of a statement can be used to determine the outcome of an algorithm. We shall learn how to build new statements, called *compound statements*, from old ones. This is especially useful in computer programs, where there are a small number of relatively simple statements we can use to control our algorithms. By building compound statements from these simple ones, we can achieve quite complex goals.