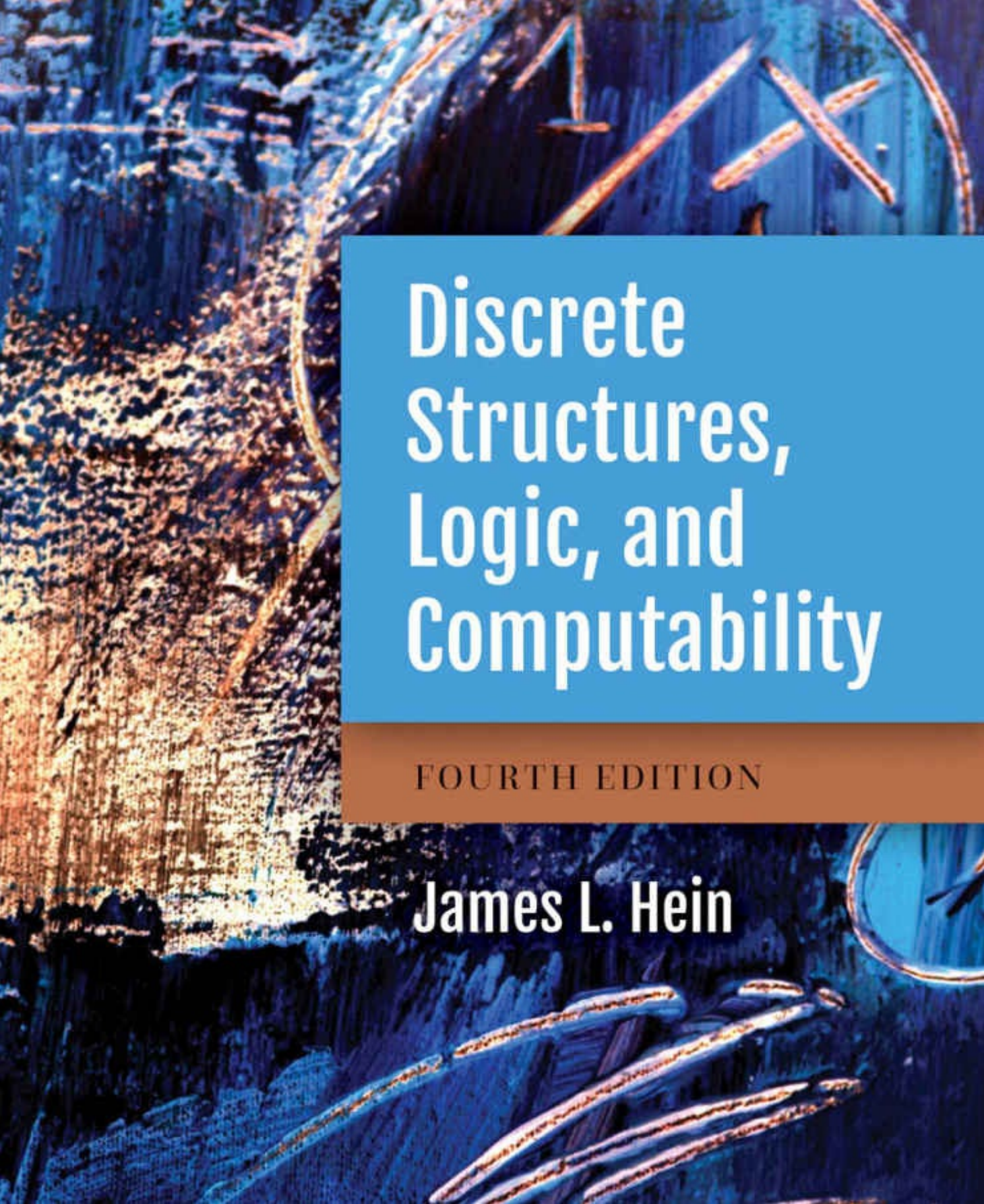
Discrete Structures, Logic, and Computability
FOURTH EDITION
James L. Hein

Discrete Structures, Logic, and Computability

FOURTH EDITION

James L. Hein
Professor Emeritus
Portland State University

World Headquarters
Jones & Bartlett Learning
5 Wall Street
Burlington, MA 01803
978-443-5000
info@jblearning.com
www.jblearning.com

Jones & Bartlett Learning books and products are available through most bookstores and online booksellers. To contact Jones & Bartlett Learning directly, call 800-832-0034, fax 978-443-8000, or visit our website, www.jblearning.com.

Substantial discounts on bulk quantities of Jones & Bartlett Learning publications are available to corporations, professional associations, and other qualified organizations. For details and specific discount information, contact the special sales department at Jones & Bartlett Learning via the above contact information or send an email to specialsales@jblearning.com.

Production Credits
VP, Executive Publisher: David D. Cella
Publisher: Cathy L. Esperti
Acquisitions Editor: Laura Pagluica
Editorial Assistant: Taylor Ferracane
Production Editor: Sara Kelly
Senior Marketing Manager: Andrea DeFronzo
Manufacturing and Inventory Control: Therese Connell
Composition: Cenveo Publisher Services
Cover Design: Kristin E. Parker
Rights and Media Research Coordinator: Merideth Tumasz
Media Development Assistant: Shannon Sheehan
Cover Image: © Marco Andras/Orange Stock RF/age fotostock
Printing and Binding: RR Donnelley
Cover printing: RR Donnelley

To order this product, use ISBN: 978-1-284-07040-8

Library of Congress Cataloging-in-Publication Data
Hein, James L.
Discrete structures, logic, and computability / James L. Hein
—Fourth edition.
pages ; cm
Includes bibliographical references and index.
ISBN 978-1-284-09986-7 (casebound)
1. Computer science—Mathematics. 2. Logic, Symbolic and mathematical. 3. Computable functions. I. Title.
QA76.9.M35.H44 2016

005.1’15–dc23

2015019684

6048

Printed in the United States of America

19 18 17 16 15 10 9 8 7 6 5 4 3 2 1

Contents

Index

Preface

The last thing one discovers in writing a book
is what to put first.
—Blaise Pascal (1623–1662)

This book is written for the prospective computer scientist, computer engineer, or applied mathematician who wants to learn the ideas that underlie computer science. The topics come from the fields of mathematics, logic, and computer science itself. I have attempted to give elementary introductions to those ideas and techniques that are necessary to understand and practice the art and science of computing. This fourth edition of the book contains all the topics for discrete structures listed in *Computer Science Curricula 2013* by the ACM/IEEE Joint Task Force on Computing Curricula.

Structure and Method

The structure of the fourth edition continues to support the spiral (i.e., iterative or nonlinear) method for learning. The spiral method is a "just in time" approach. In other words, start by introducing just enough basic information about a topic so that students can do something with it. Then revisit the topic whenever new skills or knowledge about the topic are needed for students to solve problems in other topics that have been introduced in the same way. The process continues as much as possible for each topic.

Topics that are revisited with the spiral approach include logic, sets, relations, graphs, counting, number theory, cryptology, algorithm analysis, complexity, algebra, languages, and machines. Therefore, many traditional topics are dispersed throughout the text to places where they fit naturally with the techniques under discussion.

The coverage of logic is much more extensive than in other current books at this level. Logic is of fundamental importance in computer science—not only for its use in problem solving, but also for its use in formal specification of programs, for its formal verification of programs, and for its growing use in areas such as databases, artificial intelligence, robotics, and automatic reasoning systems. Logic is covered in a spiral manner. For example, informal proof techniques are introduced in the first section of Chapter 1. Then we use informal logic without much comment until Chapter 4, where inductive proof techniques are presented. After the informal use of logic is well in hand, we move to the formal aspects of logic in Chapters 6 and 7, where equivalence proofs and rule-based proofs are introduced. Formal logic is applied to proving correctness properties of programs in Chapter 8, where we also introduce higher forms of logic and automatic reasoning.

The coverage of algebraic structures differs from that in other texts. In Chapter 9 we give elementary introductions to algebras and algebraic techniques that apply directly to computer science. In addition to the traditional topic of Boolean algebra, we introduce congruences with applications to cryptology, abstract data types, relational algebra for relational databases, functional algebra for reasoning about programs, and morphisms. In Chapter 11 we introduce the algebra of regular expressions for simplifying representations of regular languages.

The computing topics of languages, automata, and computation are introduced in Chapters 11 and 12. The

last section of the book gives an elementary introduction to computational complexity.

Changes for the Fourth Edition

- Every section of the book now contains learning objectives and review questions. There are over 350 review questions in the book.
- The first section of the book on informal proof ("A Proof Primer") has been expanded to provide a wider range of proof techniques, along with simple examples of informal proofs that use the techniques.
- A new chapter on graphs has been added (Chapter 10). It expands on the introductory material contained in the first chapter.
- Two new topics, conditional independence and elementary statistics, have been added to Section 5.4 on discrete probability.
- The coverage of logic programming has been reduced, but it is still introduced as an application of resolution in Section 8.3. The coverage of parsing algorithms has been dropped.
- Over 125 examples with named headings have been added. There are now more than 550 such headings that contain over 650 individual examples. In addition to the examples that occur under named headings, the book now has more than 1000 examples that occur within the prose, most of which are introduced by the phrase "For example."
- More than 300 new exercises have been added so that the book now contains over 2250 exercises. Answers are provided for about half of the exercises; these exercises are identified with bold numbers.

I hope that this edition has no errors. But I do wish to apologize in advance for any errors found in the book. I would appreciate hearing about them. As always, we should read the printed word with some skepticism.

Note to the Student

Most problems in computer science involve one of the following five questions:

- Can the problem be solved by a computer program?
- If not, can you modify the problem so that it can be solved by a program?
- If so, can you write a program to solve the problem?
- Can you convince another person that your program is correct?
- Can you convince another person that your program is efficient?

One goal of the book is that you obtain a better understanding of these questions, together with a better ability to answer them. The book's ultimate goal is that you gain self-reliance and confidence in your own ability to solve problems, just like the self-reliance and confidence you have in your ability to ride a bike.

Instructor and Student Resources

The companion website for the book, go.jblearning.com/Hein4e, contains additional information for the instructor, including an Instructor's Solutions Manual, slides in PowerPoint format, and Sample Exam Questions. For the student, each new copy of the textbook is equipped with an access code for the

accompanying Companion Website. The Companion Website provides access to a Student Study Guide and a Lab Book of experiments that use a free open-source mathematics software system.

Using the Book

The book can be read by anyone with a good background in high school mathematics. So it could be used at the freshman level or at the advanced high school level. As with most books, there are some dependencies among the topics. But you should feel free to jump into the book at whatever topic suits your fancy and then refer back to unfamiliar definitions if necessary. We should note that some approximation methods in Chapter 5 rely on elementary calculus, but the details can be skipped over without losing the main points.

Acknowledgments

Many people have influenced the content and form of the book. Thanks go especially to the students and teachers who have kept me busy with questions, suggestions, and criticisms. A special thanks to Professor Yasushi Kambayashi for his suggestions and for his work on the Japanese version of the book.

J.L.H.

chapter 1
Elementary Notions and Notations

‘Excellent!’ I cried. ‘Elementary,’ said he.
—Watson in *The Crooked Man*
by Arthur Conan Doyle (1859–1930)

To communicate, we sometimes need to agree on the meaning of certain terms. If the same idea is mentioned several times in a discussion, we often replace it with some shorthand notation. The choice of notation can help us avoid wordiness and ambiguity, and it can help us achieve conciseness and clarity in our written and oral expression.

Since much of our communication involves reasoning about things, we’ll begin this chapter with a discussion about the notions of informal proof. The rest of the chapter is devoted to introducing the basic notions and notations for sets, ordered structures, graphs, and trees. The treatment here is introductory in nature, and we’ll expand on these ideas in later chapters as the need arises.

1.1 A Proof Primer

For our purposes an ***informal proof*** is a demonstration that some statement is true. We normally communicate an informal proof in an English-like language that mixes everyday English with symbols that appear in the statement to be proved. In the following paragraphs we’ll discuss some basic techniques for doing informal proofs. These techniques will come in handy in trying to understand someone’s proof or in trying to construct a proof of your own, so keep them in your mental toolkit.

We’ll start off with a short refresher on logical statements followed by a short discussion about numbers. This will give us something to talk about when we look at examples of informal proof techniques.

S	not S
T	F
F	T

Figure 1.1.1 Truth table.

Statements and Truth Tables

For this primer we’ll consider only statements that are either true or false. To indicate that a statement is true (i.e., it is a truth), we assign it the ***truth value*** T (or True). Similarly, to indicate that a statement is false (i.e.,

it is a falsity) we assign it the ***truth value*** F (or False). We'll start by discussing some familiar ways to structure such statements.

Negation

If *S* represents some statement, then the ***negation*** of *S* is the statement "not *S*," whose truth value is opposite that of *S*. We can represent this relationship with a ***truth table*** in which each row gives a possible truth value for *S* and the corresponding truth value for not *S*. The truth table for negation is given in Figure 1.1.1.

We often paraphrase the negation of a statement to make it more understandable. For example, to negate the statement "Earth is a star," we normally say, "Earth is not a star," or "It is not the case that Earth is a star," rather than "Not Earth is a star."

We should also observe that negation relates statements about ***every*** case with statements about ***some*** case. For example, the statement "Not every planet has a moon" has the same meaning as "Some planet does not have a moon." Similarly, the statement "It is not the case that some planet is a star" has the same meaning as "Every planet is not a star."

Conjunction and Disjunction

The ***conjunction*** of *A* and *B* is the statement "*A* and *B*," which is true when both *A* and *B* are true. The ***disjunction*** of *A* and *B* is the statement "*A* or *B*," which is true if either or both of *A* and *B* are true. The truth tables for conjunction and disjunction are given in Figure 1.1.2.

A	*B*	*A* and *B*	*A* or *B*
T	T	T	T
T	F	F	T
F	T	F	T
F	F	F	F

Figure 1.1.2 Truth tables.

A	*B*	if *A* then *B*
T	T	T
T	F	F
F	T	T
F	F	T

Figure 1.1.3 Truth table.

Sometimes we paraphrase conjunctions and disjunctions. For example, instead of "Earth is a planet and Mars is a planet," we might write "Earth and Mars are planets." Instead of "*x* is positive or *y* is positive," we might write "Either *x* or *y* is positive."

Conditional Statements

Many statements are written in the general form "If *A* then *B*," where *A* and *B* are also statements. Such a statement is called a ***conditional statement*** in which *A* is the ***antecedent*** and *B* is the ***consequent***. The statement can also be read as "*A* ***implies*** *B*." Sometimes, but not often, it is read as "*A* is a ***sufficient condition*** for *B*," or "*B* is a ***necessary condition*** for *A*." The truth table for a conditional statement is contained in Figure 1.1.3.

Let's make a few comments about this table. Notice that the conditional is false only when the antecedent is true and the consequent is false. It's true in the other three cases. The conditional truth table gives some people fits because they interpret "If *A* then *B*" to mean "*B* can be proved from *A*," which assumes that *A* and *B* are related in some way. But we've all heard statements like "If the moon is made of green cheese, then 1 = 2." We nod our heads and agree that the statement is true, even though there is no relationship between the antecedent and consequent. Similarly, we shake our heads and don't agree with a statement like "If 1 = 1, then the moon is made of green cheese."

When the antecedent of a conditional is false, we say that the conditional is ***vacuously true***. For example, the statement "If 1 = 2, then 39 = 12" is vacuously true because the antecedent is false. If the consequent is true, we say that the conditional is ***trivially true***. For example, the statement "If 1 = 2, then 2 + 2 = 4" is trivially true because the consequent is true. We leave it to the reader to convince at least one person that the conditional truth table is defined properly.

The ***converse*** of "If *A* then *B*" is "If *B* then *A*" The converse of a statement does not always have the same truth value. For example, suppose *x* represents some living thing; then the following statement is true: "If *x* is an ant, then *x* has six legs." The converse of the statement is "If *x* has six legs, then *x* is an ant." This statement is false because, for example, bees have six legs.

Equivalent Statements

Sometimes it's convenient to write a statement in a different form but with the same truth value. Two statements are said to be ***equivalent*** if they have the same truth value for any assignment of truth values to the variables that occur in the statements.

We can combine negation with either conjunction or disjunction to obtain the following pairs of equivalent statements.

"not (A and B)" is equivalent to "(not A) or (not B)."

"not (A or B)" is equivalent to "(not A) and (not B)."

For example, the statement "not ($x > 0$ and $y > 0$)" is equivalent to the statement "$x \leq 0$ or $y \leq 0$." The statement "not ($x > 0$ or $y > 0$)" is equivalent to the statement "$x \leq 0$ and $y \leq 0$."

Conjunctions and disjunctions distribute over each other in the following sense:

"A and (B or C)" is equivalent to "(A and B) or (A and C)."

"A or (B and C)" is equivalent to "(A or B) and (A or C)."

For example, the following two statements are equivalent:

"I ate an apple and (I ate cake or I ate ice cream)."

"(I ate an apple and I ate cake) or (I ate an apple and I ate ice cream)."

The following two statements are also equivalent:

"I ate an apple or (I ate cake and I ate ice cream)."

"(I ate an apple or I ate a cake) and (I ate an apple or I ate ice cream)."

Equivalences Involving Conditionals

The ***contrapositive*** of "If A then B" is the equivalent statement "If not B then not A." For example, the following two statements are equivalent:

"If x is an ant, then x has six legs."

"If x does not have six legs, then x is not an ant."

We can also express a conditional statement without using if-then. The statement "If A then B" is equivalent to "(not A) or B." For example, the following two statements are equivalent:

"If x is an ant, then x has six legs."

"x is not an ant or x has six legs."

Since we can express a conditional in terms of negation and disjunction, it follows that we can express the negation of a conditional in terms of negation and conjunction. The statement "not (If A then B)" is equivalent to "A and (not B)." For example, the following two statements are equivalent:

"It is not true that if Earth is a planet, then Earth is a star."

"Earth is a planet and Earth is not a star."

Let's summarize the equivalences that we have discussed. Each row of the table in Figure 1.1.4 contains two equivalent statements. In other words, the truth tables for each pair of statements are identical.

Table of Equivalent Statements	
not (*A* and *B*)	(not *A*) or (not *B*)
not (*A* or *B*)	(not *A*) and (not *B*)
A and (*B* or *C*)	(*A* and *B*) or (*A* and *C*)
A or (*B* and *C*)	(*A* or *B*) and (*A* or *C*)
if *A* then *B*	if not *B* then not *A*
if *A* then *B*	(not *A*) or *B*
not (if *A* then *B*)	*A* and (not *B*)

Figure 1.1.4 Equivalent statements.

Something to Talk About

To discuss proof techniques, we need something to talk about when giving sample proofs. Since numbers are familiar to everyone, that's what we'll talk about. But to make sure that we all start on the same page, we'll review a little terminology.

The numbers that we'll be discussing are called ***integers***, and we can list them as follows:

$$\ldots, -4, -3, -2, -1, 0, 1, 2, 3, 4, \ldots .$$

The integers in the following list are called ***even integers***:

$$\ldots, -4, -2, 0, 2, 4, \ldots .$$

The integers in the following list are called ***odd integers***:

$$\ldots, -3, -1, 1, 3, \ldots .$$

So every integer is either even or odd but not both. In fact, every even integer has the form $2n$ for some integer n. Similarly, every odd integer has the form $2n + 1$ for some integer n.

Divisibility and Prime Numbers

An integer d ***divides*** an integer n if $d \neq 0$ and there is an integer k such that $n = dk$. For example, 3 divides 18 because we can write $18 = (3)(6)$. But 5 does not divide 18 because there is no integer k such that $18 = 5k$. The following list shows all the divisors of 18.

$$-18, -9, -6, -3, -2, -1, 1, 2, 3, 6, 9, 18.$$

Some alternative words for d divides n are d ***is a divisor of*** n or n ***is divisible by*** d. We often denote the fact that d divides n with the following shorthand notation:

$$d|n.$$

For example, we have $-9|9$, $-3|9$, $-1|9$, $1|9$, $3|9$, and $9|9$. Here are two properties of divisibility that we'll record for future use.

Divisibility Properties **(1.1.1)**

a. If $d|a$ and $a|b$, then $d|b$.

b. If $d|a$ and $d|b$, then $d|(ax + by)$ for any integers x and y.

An integer $p > 1$ is called a ***prime number*** if 1 and p are its only positive divisors. For example, the first eight prime numbers are

$$2, 3, 5, 7, 11, 13, 17, 19.$$

Prime numbers have many important properties, and they have many applications in computer science. But for now, all we need to know is the definition of *prime*.

Proof Techniques

Now that we have something to talk about, we'll discuss some fundamental proof techniques and give some sample proofs for each technique.

Proof by Exhaustive Checking

When a statement asserts that each of a finite number of things has a certain property, then we might be able to prove the statement by ***exhaustive checking***, where we check that each thing has the stated property.

Example 1 Exhaustive Checking

Suppose someone says, "If n is an integer and $2 \leq n \leq 7$, then $n^2 + 2$ is not divisible by 4." For $2 \leq n \leq 7$, the corresponding values of $n^2 + 2$ are

6, 11, 18, 27, 38, 51.

We can check that these numbers are not divisible by 4. For another example, suppose someone says, "If n is an integer and $2 \le n \le 500$, then $n^2 + 2$ is not divisible by 4." Again, this statement can be proved by exhaustive checking, but perhaps by a computer rather than by a person.

Exhaustive checking cannot be used to prove a statement that requires infinitely many things to check. For example, consider the statement, "If n is an integer, then $n^2 + 2$ is not divisible by 4." This statement is true, but there are infinitely many things to check. So another proof technique will be required. We'll get to it after a few more paragraphs.

An example that proves a statement false is often called a ***counterexample***. Sometimes counterexamples can be found by exhaustive checking. For example, consider the statement, "Every odd number greater than 1 that is not prime has the form $2 + p$ for some prime p." We can observe that the statement is false because 27 is a counterexample.

Conditional Proof

Many statements that we wish to prove are in conditional form or can be rephrased in conditional form. The ***direct approach*** to proving a conditional of the form "if A then B" starts with the assumption that the antecedent A is true. The next step is to find a statement that is implied by the assumption or known facts. Each step proceeds in this fashion to find a statement that is implied by any of the previous statements or known facts. The ***conditional proof*** ends when the consequent B is reached.

Example 2 An Even Sum

We'll prove that the sum of any two odd integers is an even integer. We can rephrase the statement in the following conditional form:

If x and y are odd, then $x + y$ is even.

Proof: Assume that x and y are odd integers. It follows that x and y can be written in the form $x = 2k + 1$ and $y = 2m + 1$ for some integers k and m. Now, substitute for x and y in the sum $x + y$ to obtain

$$x + y = (2k + 1) + (2m + 1) = 2k + 2m + 2 = 2(k + m + 1).$$

Since the expression on the right-hand side contains 2 as a factor, it represents an even integer. QED.

Example 3 An Even Product

We'll prove the following statement about integers:

If x is even, then xy is even for any integer y.

Proof: Assume that x is even. Then $x = 2k$ for some integer k. If y is any integer, then we can write $xy = (2k)y = 2(ky)$, which is the expression for an even integer. QED.

Example 4 A Proof with Two Cases

We'll prove the following statement about integers:

If y is odd, then $x(x + y)$ is even for any integer x.

Proof: Assume that y is odd. Let x be an arbitrary integer. Since x is either even or odd, we'll split the proof into two cases. Case 1: If x is even, then the product $x(x + y)$ is even by Example 3. Case 2: If x is odd, then the sum $x + y$ is even by Example 2. It follows that the product $x(x + y)$ is even by Example 3. So, in either case, $x(x + y)$ is even. Therefore, the statement is true. QED.

Example 5 A Divisibility Proof

We'll prove the following statement about divisibility of integers:

If $x|(x + 1)y$, then $x|y$.

Proof: Assume that $x|(x + 1)y$. Then $(x + 1)y = xk$ for some integer k. The equation can be written as $xy + y = xk$. Subtract xy from both sides to obtain

$$y = xk - xy = x(k - y).$$

This equation tells us that $x|y$. QED.

Example 6 A Disjunctive Antecedent

Suppose we want to prove the following statement about integers:

If x is even or y is odd, then $x + xy$ is even.

How do we handle the disjunction in the antecedent? We can observe the following equivalence by checking truth tables:

"If (A or B) then C" is equivalent to "(If A then C) and (If B then C)."

The statement can be proved by proving the two statements "If A then C" and "If B then C." So we'll prove the following two statements:

If x is even, then $x + xy$ is even.

If y is odd, then $x + xy$ is even.

Proof: Assume that x is even. Then $x = 2k$ for some integer k. It follows that

$$x + xy = 2k + 2ky = 2(k + ky).$$

This equation tells us that $x + xy$ is even. Next, assume that y is odd. Then $y = 2k + 1$ for some integer k. It follows that

$$x + xy = x + x(2k + 1) = x + 2kx + x = 2(x + 2k).$$

This equation tells us that $x + xy$ is even. QED.

Example 7 Proving a Contrapositive

Suppose we want to prove the following statement about the integers:

If xy is odd, then x is odd and y is odd.

If we assume that xy is odd, then we can write $xy = 2k + 1$ for some integer k. This information doesn't seem all that helpful in moving us to the conclusion. Let's consider another way to prove the statement.

Recall that a conditional statement "if A then B" and its contrapositive "if not B then not A" are equivalent. So, a proof of one is also a proof of the other. Sometimes the contrapositive is easier to prove, and this example is such a case. We'll prove the following contrapositive form of the statement:

If x is even or y is even, then xy is even.

Proof: Because the antecedent is a disjunction, it follows from Example 6 that we need two proofs. First, assume that x is even. Then we can use the result of Example 3 to conclude that xy is even. Next, assume that y is even. Again, by Example 3, we can conclude that xy ($= yx$) is even. Therefore, the given statement is true. QED.

Example 8 A Disjunction

Suppose we want to prove the following statement about integers:

$2 \mid 3x$ or x is odd.

It's always nice to start a proof with an assumption. But what can we assume? We can observe the following equivalence by checking truth tables:

"A or B" is equivalent to "If (not A) then B."

So it suffices to prove "If (not A) then B." In other words, assume that A is false and give an argument showing that B is true. So, we'll prove the following statement:

If 2 does not divide $3x$, then x is odd.

Proof: Assume that 2 does not divide $3x$. Then $3x \neq 2k$ for every integer k. This says that $3x$ is not even. Therefore, $3x$ is odd, and it follows from Example 7 that x is odd. QED.

Note: Since "A or B" is equivalent to "B or A," we could also prove the statement by assuming that B is false and giving an argument that A is true. It's nice to have a choice, since one proof might be easier. We'll prove the following alternative statement:

If x is even, then $2 \mid 3x$.