Part I

Proofs

Mathematical Proofs

A proof is a method of establishing truth. What constitutes a proof differs among fields.

- *Legal* truth is decided by a jury based on allowable evidence presented at trial.
- *Authoritative* truth is specified by a trusted person or organization.
- *Scientific* truth[1] is confirmed by experiment.
- *Probable* truth is established by statistical analysis of sample data.
- *Philosophical* proof involves careful exposition and persuasion typically based on a series of small, plausible arguments. The best example begins with "Cogito ergo sum," a Latin sentence that translates as "I think, therefore I am." It comes from the beginning of a 17th century essay by the mathematician/philospher, René Descartes, and it is one of the most famous quotes in the world: do a web search on the phrase and you will be flooded with hits.

  Deducing your existence from the fact that you're thinking about your existence is a pretty cool and persuasive-sounding first axiom. However, with just a few more lines of argument in this vein, Descartes goes on to conclude that there is an infinitely beneficent God. Whether or not you believe in a beneficent God, you'll probably agree that any very short proof of God's existence is bound to be far-fetched. So even in masterful hands, this approach is not reliable.

Mathematics also has a specific notion of "proof."

Definition. A *formal proof* of a *proposition* is a chain of *logical deductions* leading to the proposition from a base set of *axioms*.

The three key ideas in this definition are highlighted: proposition, logical deduction, and axiom. These three ideas are explained in Chapter 1, and Chapter 2 describes some basic ways of organizing proofs.

0.0.1 Problems

Class Problems

Problem 0.1.
Identify exactly where the bugs are in each of the following bogus proofs.[2]

[1] Actually, only scientific *falsehood* can be demonstrated by an experiment —when the experiment fails to behave as predicted. But no amount of experiment can confirm that the *next* experiment won't fail. For this reason, scientists rarely speak of truth, but rather of *theories* that accurately predict past, and anticipated future, experiments.

[2] From Stueben, Michael and Diane Sandford. *Twenty Years Before the Blackboard*, Mathematical Association of America, ©1998.

(a) Bogus Claim: $1/8 > 1/4$.

Bogus proof.

$$\begin{aligned} 3 &> 2 \\ 3\log_{10}(1/2) &> 2\log_{10}(1/2) \\ \log_{10}(1/2)^3 &> \log_{10}(1/2)^2 \\ (1/2)^3 &> (1/2)^2, \end{aligned}$$

and the claim now follows by the rules for multiplying fractions. ■

(b) *Bogus proof*: $1¢ = \$0.01 = (\$0.1)^2 = (10¢)^2 = 100¢ = \$1$. ■

(c) Bogus Claim: If a and b are two equal real numbers, then $a = 0$.

Bogus proof.

$$\begin{aligned} a &= b \\ a^2 &= ab \\ a^2 - b^2 &= ab - b^2 \\ (a-b)(a+b) &= (a-b)b \\ a+b &= b \\ a &= 0. \end{aligned}$$

■

Problem 0.2.
It's a fact that the Arithmetic Mean is at least as large the Geometric Mean, namely,

$$\frac{a+b}{2} \geq \sqrt{ab}$$

for all nonnegative real numbers a and b. But there's something objectionable about the following proof of this fact. What's the objection, and how would you fix it?

Bogus proof.

$$\begin{aligned} \frac{a+b}{2} &\overset{?}{\geq} \sqrt{ab}, && \text{so} \\ a+b &\overset{?}{\geq} 2\sqrt{ab}, && \text{so} \\ a^2 + 2ab + b^2 &\overset{?}{\geq} 4ab, && \text{so} \\ a^2 - 2ab + b^2 &\overset{?}{\geq} 0, && \text{so} \\ (a-b)^2 &\geq 0 && \text{which we know is true.} \end{aligned}$$

The last statement is true because $a - b$ is a real number, and the square of a real number is never negative. This proves the claim. ■

Problem 0.3.
Albert announces to his class that he plans to surprise them with a quiz sometime next week.

His students first wonder if the quiz could be on Friday of next next. They reason that it can't: if Albert didn't give the quiz *before* Friday, then by midnight Thursday, they would know the quiz had to be on Friday, and so the quiz wouldn't be a surprise any more.

Next the students wonder whether Albert could give the surprise quiz Thursday. They observe that if the quiz wasn't given *before* Thursday, it would have to be given *on* the Thursday, since they already know it can't be given on Friday. But having figured that out, it wouldn't be a surprise if the quiz was on Thursday either. Similarly, the students reason that the quiz can't be on Wednesday, Tuesday, or Monday. Namely, it's impossible for Albert to give a surprise quiz next week. All the students now relax, having concluded that Albert must have been bluffing.

And since no one expects the quiz, that's why, when Albert gives it on Tuesday next week, it really is a surprise!

What do you think is wrong with the students' reasoning?

0.1 Propositions

Definition. A *proposition* is a statement that is either true or false.

This definition sounds very general, but it does exclude sentences such as, "Wherefore art thou Romeo?" and "Give me an A!". But not all propositions are mathematical. For example, "Albert's wife's name is 'Irene' " happens to be true, and could be proved with legal documents and testimony of their children, but it's not a mathematical statement.

Mathematically meaningful propositions must be about well-defined mathematical objects like numbers, sets, functions, relations, etc., and they must be stated using mathematically precise language. We can illustrate this with a few examples.

Proposition 0.1.1. *2 + 3 = 5.*

This proposition is true.

A *prime* is an integer greater than one that is not divisible by any integer greater than 1 besides itself, for example, 2, 3, 5, 7, 11, ...

Proposition 0.1.2. *For every nonnegative integer, n, the value of $n^2 + n + 41$ is prime.*

Let's try some numerical experimentation to check this proposition. Let [3]

$$p(n) ::= n^2 + n + 41. \tag{1}$$

We begin with $p(0) = 41$ which is prime. $p(1) = 43$ which is prime. $p(2) = 47$ which is prime. $p(3) = 53$ which is prime. $\ldots p(20) = 461$ which is prime. Hmmm, starts to look like a plausible claim. In fact we can keep checking through $n = 39$ and confirm that $p(39) = 1601$ is prime.

But $p(40) = 40^2 + 40 + 41 = 41 \cdot 41$, which is not prime. So it's not true that the expression is prime *for all* nonnegative integers. The point is that in general you can't check a claim about an infinite set by checking a finite set of its elements, no matter how large the finite set.

By the way, propositions like this about *all* numbers or other things are so common that there is a special notation for it. With this notation, Proposition 1.5.1 would be

$$\forall n \in \mathbb{N}.\ p(n) \text{ is prime.} \tag{2}$$

Here the symbol $\forall$ is read "for all". The symbol $\mathbb{N}$ stands for the set of *nonnegative integers*, namely, 0, 1, 2, 3, . . . (ask your TA for the complete list). The symbol "$\in$" is read as "is a member of" or simply as "is in". The period after the $\mathbb{N}$ is just a separator between phrases.

Here are two even more extreme examples:

Proposition 0.1.3. *$a^4 + b^4 + c^4 = d^4$ has no solution when a, b, c, d are positive integers.*

Euler (pronounced "oiler") conjectured this in 1769. But the proposition was proven false 218 years later by Noam Elkies at a liberal arts school up Mass Ave. The solution he found was $a = 95800, b = 217519, c = 414560, d = 422481$.

In logical notation, Proposition 1.5.2 could be written,

$$\forall a \in \mathbb{Z}^+\ \forall b \in \mathbb{Z}^+\ \forall c \in \mathbb{Z}^+\ \forall d \in \mathbb{Z}^+.\ a^4 + b^4 + c^4 \neq d^4.$$

Here, $\mathbb{Z}^+$ is a symbol for the positive integers. Strings of $\forall$'s like this are usually abbreviated for easier reading:

$$\forall a, b, c, d \in \mathbb{Z}^+.\ a^4 + b^4 + c^4 \neq d^4.$$

Proposition 0.1.4. *$313(x^3 + y^3) = z^3$ has no solution when $x, y, z \in \mathbb{Z}^+$.*

This proposition is also false, but the smallest counterexample has more than 1000 digits!

Proposition 0.1.5. *Every map can be colored with 4 colors so that adjacent[4] regions have different colors.*

[3]The symbol ::= means "equal by definition." It's always ok to simply write "=" instead of ::=, but reminding the reader that an equality holds by definition can be helpful.

[4]Two regions are adjacent only when they share a boundary segment of positive length. They are not considered to be adjacent if their boundaries meet only at a few points.

This proposition is true and is known as the *"Four-Color Theorem"*. However, there have been many incorrect proofs, including one that stood for 10 years in the late 19th century before the mistake was found. An extremely laborious proof was finally found in 1976 by mathematicians Appel and Haken, who used a complex computer program to categorize the four-colorable maps; the program left a couple of thousand maps uncategorized, and these were checked by hand by Haken and his assistants—including his 15-year-old daughter. There was a lot of debate about whether this was a legitimate proof: the proof was too big to be checked without a computer, and no one could guarantee that the computer calculated correctly, nor did anyone have the energy to recheck the four-colorings of thousands of maps that were done by hand. Finally, about five years ago, a mostly intelligible proof of the Four-Color Theorem was found, though a computer is still needed to check colorability of several hundred special maps (see
`http://www.math.gatech.edu/~thomas/FC/fourcolor.html`). [5]

Proposition 0.1.6 (Goldbach)**.** *Every even integer greater than 2 is the sum of two primes.*

No one knows whether this proposition is true or false. It is known as *Goldbach's Conjecture*, and dates back to 1742.

For a computer scientist, some of the most important things to prove are the "correctness" programs and systems —whether a program or system does what it's supposed to. Programs are notoriously buggy, and there's a growing community of researchers and practitioners trying to find ways to prove program correctness. These efforts have been successful enough in the case of CPU chips that they are now routinely used by leading chip manufacturers to prove chip correctness and avoid mistakes like the notorious Intel division bug in the 1990's.

Developing mathematical methods to verify programs and systems remains an active research area. We'll consider some of these methods later in the course.

0.2 Predicates

A *predicate* is a proposition whose truth depends on the value of one or more variables. Most of the propostions above were defined in terms of predicates. For example,

$$\text{“}n \text{ is a perfect square”}$$

is a predicate whose truth depends on the value of n. The predicate is true for $n = 4$ since four is a perfect square, but false for $n = 5$ since five is not a perfect square.

Like other propositions, predicates are often named with a letter. Furthermore, a function-like notation is used to denote a predicate supplied with specific vari-

[5]The story of the Four-Color Proof is told in a well-reviewed popular (non-technical) book: "Four Colors Suffice. How the Map Problem was Solved." *Robin Wilson*. Princeton Univ. Press, 2003, 276pp. ISBN 0-691-11533-8.

able values. For example, we might name our earlier predicate P:

$$P(n) ::= \text{“}n \text{ is a perfect square”}$$

Now $P(4)$ is true, and $P(5)$ is false.

This notation for predicates is confusingly similar to ordinary function notation. If P is a predicate, then $P(n)$ is either *true* or *false*, depending on the value of n. On the other hand, if p is an ordinary function, like $n^2 + 1$, then $p(n)$ is a *numerical quantity*. **Don't confuse these two!**

0.3 The Axiomatic Method

The standard procedure for establishing truth in mathematics was invented by Euclid, a mathematician working in Alexandria, Egypt around 300 BC. His idea was to begin with five *assumptions* about geometry, which seemed undeniable based on direct experience. (For example, "There is a straight line segment between every pair of points.) Propositions like these that are simply accepted as true are called *axioms*.

Starting from these axioms, Euclid established the truth of many additional propositions by providing "proofs". A *proof* is a sequence of logical deductions from axioms and previously-proved statements that concludes with the proposition in question. You probably wrote many proofs in high school geometry class, and you'll see a lot more in this course.

There are several common terms for a proposition that has been proved. The different terms hint at the role of the proposition within a larger body of work.

- Important propositions are called *theorems*.
- A *lemma* is a preliminary proposition useful for proving later propositions.
- A *corollary* is a proposition that follows in just a few logical steps from a theorem.

The definitions are not precise. In fact, sometimes a good lemma turns out to be far more important than the theorem it was originally used to prove.

Euclid's axiom-and-proof approach, now called the *axiomatic method*, is the foundation for mathematics today. In fact, just a handful of axioms, called the axioms Zermelo-Frankel with Choice (*ZFC*), together with a few logical deduction rules, appear to be sufficient to derive essentially all of mathematics. We'll examine these in Chapter 5.

0.4 Our Axioms

The ZFC axioms are important in studying and justifying the foundations of mathematics, but for practical purposes, they are much too primitive. Proving theorems in ZFC is a little like writing programs in byte code instead of a full-fledged programming language —by one reckoning, a formal proof in ZFC that $2 + 2 = 4$ requires more than 20,000 steps! So instead of starting with ZFC, we're going to take a *huge* set of axioms as our foundation: we'll accept all familiar facts from high school math!

This will give us a quick launch, but you may find this imprecise specification of the axioms troubling at times. For example, in the midst of a proof, you may find yourself wondering, "Must I prove this little fact or can I take it as an axiom?" Feel free to ask for guidance, but really there is no absolute answer. Just be up front about what you're assuming, and don't try to evade homework and exam problems by declaring everything an axiom!

0.4.1 Logical Deductions

Logical deductions or *inference rules* are used to prove new propositions using previously proved ones.

A fundamental inference rule is *modus ponens*. This rule says that a proof of P together with a proof that P IMPLIES Q is a proof of Q.

Inference rules are sometimes written in a funny notation. For example, *modus ponens* is written:

Rule.

$$\frac{P, \quad P \text{ IMPLIES } Q}{Q}$$

When the statements above the line, called the *antecedents*, are proved, then we can consider the statement below the line, called the *conclusion* or *consequent*, to also be proved.

A key requirement of an inference rule is that it must be *sound*: any assignment of truth values that makes all the antecedents true must also make the consequent true. So if we start off with true axioms and apply sound inference rules, everything we prove will also be true.

There are many other natural, sound inference rules, for example:

Rule.

$$\frac{P \text{ IMPLIES } Q, \quad Q \text{ IMPLIES } R}{P \text{ IMPLIES } R}$$

Rule.

$$\frac{\text{NOT}(P) \text{ IMPLIES NOT}(Q)}{Q \text{ IMPLIES } P}$$

On the other hand,

Rule.

$$\frac{\text{NOT}(P) \text{ IMPLIES NOT}(Q)}{P \text{ IMPLIES } Q}$$

is not sound: if P is assigned **T** and Q is assigned **F**, then the antecedent is true and the consequent is not.

Note that a propositional inference rule is sound precisely when the conjunction (AND) of all its antecedents implies its consequent.

As with axioms, we will not be too formal about the set of legal inference rules. Each step in a proof should be clear and "logical"; in particular, you should state what previously proved facts are used to derive each new conclusion.

0.4.2 Patterns of Proof

In principle, a proof can be *any* sequence of logical deductions from axioms and previously proved statements that concludes with the proposition in question. This freedom in constructing a proof can seem overwhelming at first. How do you even *start* a proof?

Here's the good news: many proofs follow one of a handful of standard templates. Each proof has it own details, of course, but these templates at least provide you with an outline to fill in. We'll go through several of these standard patterns, pointing out the basic idea and common pitfalls and giving some examples. Many of these templates fit together; one may give you a top-level outline while others help you at the next level of detail. And we'll show you other, more sophisticated proof techniques later on.

The recipes below are very specific at times, telling you exactly which words to write down on your piece of paper. You're certainly free to say things your own way instead; we're just giving you something you *could* say so that you're never at a complete loss.

0.5 Proving an Implication

Propositions of the form "If P, then Q" are called *implications*. This implication is often rephrased as "P IMPLIES Q."

Here are some examples:

- (Quadratic Formula) If $ax^2 + bx + c = 0$ and $a \neq 0$, then
$$x = \left(-b \pm \sqrt{b^2 - 4ac}\right)/2a.$$
- (Goldbach's Conjecture) If n is an even integer greater than 2, then n is a sum of two primes.
- If $0 \leq x \leq 2$, then $-x^3 + 4x + 1 > 0$.

There are a couple of standard methods for proving an implication.

0.5.1 Method #1

In order to prove that P IMPLIES Q:

1. Write, "Assume P."
2. Show that Q logically follows.

Example

Theorem 0.5.1. *If* $0 \leq x \leq 2$, *then* $-x^3 + 4x + 1 > 0$.

Before we write a proof of this theorem, we have to do some scratchwork to figure out why it is true.

The inequality certainly holds for $x = 0$; then the left side is equal to 1 and $1 > 0$. As x grows, the $4x$ term (which is positive) initially seems to have greater magnitude than $-x^3$ (which is negative). For example, when $x = 1$, we have $4x = 4$, but $-x^3 = -1$ only. In fact, it looks like $-x^3$ doesn't begin to dominate until $x > 2$. So it seems the $-x^3 + 4x$ part should be nonnegative for all x between 0 and 2, which would imply that $-x^3 + 4x + 1$ is positive.

So far, so good. But we still have to replace all those "seems like" phrases with solid, logical arguments. We can get a better handle on the critical $-x^3 + 4x$ part by factoring it, which is not too hard:

$$-x^3 + 4x = x(2 - x)(2 + x)$$

Aha! For x between 0 and 2, all of the terms on the right side are nonnegative. And a product of nonnegative terms is also nonnegative. Let's organize this blizzard of observations into a clean proof.

Proof. Assume $0 \leq x \leq 2$. Then x, $2 - x$, and $2 + x$ are all nonnegative. Therefore, the product of these terms is also nonnegative. Adding 1 to this product gives a positive number, so:

$$x(2 - x)(2 + x) + 1 > 0$$

Multiplying out on the left side proves that

$$-x^3 + 4x + 1 > 0$$

as claimed. ■

There are a couple points here that apply to all proofs:

- You'll often need to do some scratchwork while you're trying to figure out the logical steps of a proof. Your scratchwork can be as disorganized as you like— full of dead-ends, strange diagrams, obscene words, whatever. But keep your scratchwork separate from your final proof, which should be clear and concise.

- Proofs typically begin with the word "Proof" and end with some sort of doohickey like □ or "q.e.d". The only purpose for these conventions is to clarify where proofs begin and end.

0.5.2 Method #2 - Prove the Contrapositive

An implication ("P IMPLIES Q") is logically equivalent to its *contrapositive*

$$\text{NOT}(Q) \text{ IMPLIES NOT}(P)$$

Proving one is as good as proving the other, and proving the contrapositive is sometimes easier than proving the original statement. If so, then you can proceed as follows:

1. Write, "We prove the contrapositive:" and then state the contrapositive.
2. Proceed as in Method #1.

Example

Theorem 0.5.2. *If r is irrational, then $\sqrt{r}$ is also irrational.*

Recall that rational numbers are equal to a ratio of integers and irrational numbers are not. So we must show that if r is *not* a ratio of integers, then $\sqrt{r}$ is also *not* a ratio of integers. That's pretty convoluted! We can eliminate both *not*'s and make the proof straightforward by considering the contrapositive instead.

Proof. We prove the contrapositive: if $\sqrt{r}$ is rational, then r is rational.

Assume that $\sqrt{r}$ is rational. Then there exist integers a and b such that:

$$\sqrt{r} = \frac{a}{b}$$

Squaring both sides gives:

$$r = \frac{a^2}{b^2}$$

Since a^2 and b^2 are integers, r is also rational. ■

0.5.3 Problems

Homework Problems

Problem 0.4.
Show that $\log_7 n$ is either an integer or irrational, where n is a positive integer. Use whatever familiar facts about integers and primes you need, but explicitly state such facts.

0.6 Proving an "If and Only If"

Many mathematical theorems assert that two statements are logically equivalent; that is, one holds if and only if the other does. Here is an example that has been known for several thousand years:

Two triangles have the same side lengths if and only if two side lengths and the angle between those sides are the same.

The phrase "if and only if" comes up so often that it is often abbreviated "iff".

0.6.1 Method #1: Prove Each Statement Implies the Other

The statement "P IFF Q" is equivalent to the two statements "P IMPLIES Q" and "Q IMPLIES P". So you can prove an "iff" by proving *two* implications:

1. Write, "We prove P implies Q and vice-versa."
2. Write, "First, we show P implies Q." Do this by one of the methods in Section 2.2.1.
3. Write, "Now, we show Q implies P." Again, do this by one of the methods in Section 2.2.1.

0.6.2 Method #2: Construct a Chain of Iffs

In order to prove that P is true iff Q is true:

1. Write, "We construct a chain of if-and-only-if implications."
2. Prove P is equivalent to a second statement which is equivalent to a third statement and so forth until you reach Q.

This method sometimes requires more ingenuity than the first, but the result can be a short, elegant proof.

Example

The *standard deviation* of a sequence of values $x_1, x_2, \ldots, x_n$ is defined to be:

$$\sqrt{\frac{(x_1-\mu)^2+(x_2-\mu)^2+\cdots+(x_n-\mu)^2}{n}} \tag{3}$$

where μ is the *mean* of the values:

$$\mu ::= \frac{x_1+x_2+\cdots+x_n}{n}$$

Theorem 0.6.1. *The standard deviation of a sequence of values $x_1, \ldots, x_n$ is zero iff all the values are equal to the mean.*

For example, the standard deviation of test scores is zero if and only if everyone scored exactly the class average.

Proof. We construct a chain of "iff" implications, starting with the statement that the standard deviation (2.1) is zero:

$$\sqrt{\frac{(x_1-\mu)^2+(x_2-\mu)^2+\cdots+(x_n-\mu)^2}{n}}=0. \tag{4}$$

Now since zero is the only number whose square root is zero, equation (2.2) holds iff

$$(x_1-\mu)^2+(x_2-\mu)^2+\cdots+(x_n-\mu)^2=0. \tag{5}$$

Now squares of real numbers are always nonnegative, so every term on the left hand side of equation (2.3) is nonnegative. This means that (2.3) holds iff

$$\text{Every term on the left hand side of (2.3) is zero.} \tag{6}$$

But a term $(x_i-\mu)^2$ is zero iff $x_i=\mu$, so (2.4) is true iff

$$\text{Every } x_i \text{ equals the mean.}$$

■

0.7 Proof by Cases

Breaking a complicated proof into cases and proving each case separately is a useful, common proof strategy. Here's an amusing example.

Let's agree that given any two people, either they have met or not. If every pair of people in a group has met, we'll call the group a *club*. If every pair of people in a group has not met, we'll call it a group of *strangers*.

Theorem. *Every collection of 6 people includes a club of 3 people or a group of 3 strangers.*

Proof. The proof is by case analysis[6]. Let x denote one of the six people. There are two cases:

1. Among 5 other people besides x, at least 3 have met x.

2. Among the 5 other people, at least 3 have not met x.

Now we have to be sure that at least one of these two cases must hold,[7] but that's easy: we've split the 5 people into two groups, those who have shaken hands with x and those who have not, so one the groups must have at least half the people.

Case 1: Suppose that at least 3 people did meet x.

This case splits into two subcases:

[6]Describing your approach at the outset helps orient the reader.

[7]Part of a case analysis argument is showing that you've covered all the cases. Often this is obvious, because the two cases are of the form "P" and "not P". However, the situation above is not stated quite so simply.

Case 1.1: No pair among those people met each other. Then these people are a group of at least 3 strangers. So the Theorem holds in this subcase.

Case 1.2: Some pair among those people have met each other. Then that pair, together with x, form a club of 3 people. So the Theorem holds in this subcase.

This implies that the Theorem holds in Case 1.

Case 2: Suppose that at least 3 people did not meet x.

This case also splits into two subcases:

Case 2.1: Every pair among those people met each other. Then these people are a club of at least 3 people. So the Theorem holds in this subcase.

Case 2.2: Some pair among those people have not met each other. Then that pair, together with x, form a group of at least 3 strangers. So the Theorem holds in this subcase.

This implies that the Theorem also holds in Case 2, and therefore holds in all cases. ■

0.7.1 Problems

Class Problems

Problem 0.5.
If we raise an irrational number to an irrational power, can the result be rational? Show that it can by considering $\sqrt{2}^{\sqrt{2}}$ and arguing by cases.

Homework Problems

Problem 0.6.
For $n = 40$, the value of polynomial $p(n) ::= n^2 + n + 41$ 1.4, is not prime, as noted in Section 1.5. But we could have predicted based on general principles that no nonconstant polynomial, $q(n)$, with integer coefficients can map each nonnegative integer into a prime number. Prove it.

Hint: Let $c ::= q(0)$ be the constant term of q. Consider two cases: c is not prime, and c is prime. In the second case, note that $q(cn)$ is a multiple of c for all $n \in \mathbb{Z}$. You may assume the familiar fact that the magnitude (absolute value) of any nonconstant polynomial, $q(n)$, grows unboundedly as n grows.

0.8 Proof by Contradiction

In a *proof by contradiction* or *indirect proof*, you show that if a proposition were false, then some false fact would be true. Since a false fact can't be true, the proposition had better not be false. That is, the proposition really must be true.

Proof by contradiction is *always* a viable approach. However, as the name suggests, indirect proofs can be a little convoluted. So direct proofs are generally preferable as a matter of clarity.

Method: In order to prove a proposition P by contradiction:

1. Write, "We use proof by contradiction."
2. Write, "Suppose P is false."
3. Deduce something known to be false (a logical contradiction).
4. Write, "This is a contradiction. Therefore, P must be true."

Example

Remember that a number is *rational* if it is equal to a ratio of integers. For example, $3.5 = 7/2$ and $0.1111\cdots = 1/9$ are rational numbers. On the other hand, we'll prove by contradiction that $\sqrt{2}$ is irrational.

Theorem 0.8.1. *$\sqrt{2}$ is irrational.*

Proof. We use proof by contradiction. Suppose the claim is false; that is, $\sqrt{2}$ is rational. Then we can write $\sqrt{2}$ as a fraction n/d in *lowest terms*.

Squaring both sides gives $2 = n^2/d^2$ and so $2d^2 = n^2$. This implies that n is a multiple of 2. Therefore n^2 must be a multiple of 4. But since $2d^2 = n^2$, we know $2d^2$ is a multiple of 4 and so d^2 is a multiple of 2. This implies that d is a multiple of 2.

So the numerator and denominator have 2 as a common factor, which contradicts the fact that n/d is in lowest terms. So $\sqrt{2}$ must be irrational. ■

0.8.1 Problems

Class Problems

Problem 0.7.
Generalize the proof of Theorem 2.2.4 that $\sqrt{2}$ is irrational. For example, how about $\sqrt[3]{2}$?

Problem 0.8.
Here is a different proof that $\sqrt{2}$ is irrational, taken from the American Mathematical Monthly, v.116, #1, Jan. 2009, p.69:

Proof. Suppose for the sake of contradiction that $\sqrt{2}$ is rational, and choose the least integer, $q > 0$, such that $\left(\sqrt{2} - 1\right) q$ is a nonnegative integer. Let $q' ::= \left(\sqrt{2} - 1\right) q$. Clearly $0 < q' < q$. But an easy computation shows that $\left(\sqrt{2} - 1\right) q'$ is a nonnegative integer, contradicting the minimality of q. ■

(a) This proof was written for an audience of college teachers, and at this point it is a little more concise than desirable. Write out a more complete version which includes an explanation of each step.

(b) Now that you have justified the steps in this proof, do you have a preference for one of these proofs over the other? Why? Discuss these questions with your teammates for a few minutes and summarize your team's answers on your whiteboard.

Problem 0.9.
Here is a generalization of Problem 2.4 that you may not have thought of:

Lemma 0.8.2. *Let the coefficients of the polynomial $a_0+a_1x+a_2x^2+\cdots+a_{n-1}x^{m-1}+x^m$ be integers. Then any real root of the polynomial is either integral or irrational.*

(a) Explain why Lemma 2.3.1 immediately implies that $\sqrt[m]{k}$ is irrational whenever k is not an mth power of some integer.

(b) Collaborate with your tablemates to write a clear, textbook quality proof of Lemma 2.3.1 on your whiteboard. (Besides clarity and correctness, textbook quality requires good English with proper punctuation. When a real textbook writer does this, it usually takes multiple revisions; if you're satisfied with your first draft, you're probably misjudging.) You may find it helpful to appeal to the following:
Lemma 0.8.3. *If a prime, p, is a factor of some power of an integer, then it is a factor of that integer.*

You may assume Lemma 2.3.2 without writing down its proof, but see if you can explain why it is true.

Homework Problems

Problem 0.10.
The fact that that there are irrational numbers a, b such that a^b is rational was proved in Problem 2.1. Unfortunately, that proof was *nonconstructive*: it didn't reveal a specific pair, a, b, with this property. But in fact, it's easy to do this: let $a ::= \sqrt{2}$ and $b ::= 2\log_2 3$.

We know $\sqrt{2}$ is irrational, and obviously $a^b = 3$. Finish the proof that this a, b pair works, by showing that $2\log_2 3$ is irrational.

0.9 *Good* Proofs in Practice

One purpose of a proof is to establish the truth of an assertion with absolute certainty. Mechanically checkable proofs of enormous length or complexity can accomplish this. But humanly intelligible proofs are the only ones that help someone understand the subject. Mathematicians generally agree that important mathematical results can't be fully understood until their proofs are understood. That is why proofs are an important part of the curriculum.

To be understandable and helpful, more is required of a proof than just logical correctness: a good proof must also be clear. Correctness and clarity usually go together; a well-written proof is more likely to be a correct proof, since mistakes are harder to hide.

In practice, the notion of proof is a moving target. Proofs in a professional research journal are generally unintelligible to all but a few experts who know all the terminology and prior results used in the proof. Conversely, proofs in the first weeks of a beginning course like 6.042 would be regarded as tediously long-winded by a professional mathematician. In fact, what we accept as a good proof later in the term will be different from what we consider good proofs in the first couple of weeks of 6.042. But even so, we can offer some general tips on writing good proofs:

State your game plan. A good proof begins by explaining the general line of reasoning, for example, "We use case analysis" or "We argue by contradiction."

Keep a linear flow. Sometimes proofs are written like mathematical mosaics, with juicy tidbits of independent reasoning sprinkled throughout. This is not good. The steps of an argument should follow one another in an intelligible order.

A proof is an essay, not a calculation. Many students initially write proofs the way they compute integrals. The result is a long sequence of expressions without explanation, making it very hard to follow. This is bad. A good proof usually looks like an essay with some equations thrown in. Use complete sentences.

Avoid excessive symbolism. Your reader is probably good at understanding words, but much less skilled at reading arcane mathematical symbols. So use words where you reasonably can.

Revise and simplify. Your readers will be grateful.

Introduce notation thoughtfully. Sometimes an argument can be greatly simplified by introducing a variable, devising a special notation, or defining a new term. But do this sparingly since you're requiring the reader to remember all that new stuff. And remember to actually *define* the meanings of new variables, terms, or notations; don't just start using them!

Structure long proofs. Long programs are usually broken into a hierarchy of smaller procedures. Long proofs are much the same. Facts needed in your proof that are easily stated, but not readily proved are best pulled out and proved in preliminary lemmas. Also, if you are repeating essentially the same argument over and over, try to capture that argument in a general lemma, which you can cite repeatedly instead.

Be wary of the "obvious". When familiar or truly obvious facts are needed in a proof, it's OK to label them as such and to not prove them. But remember that what's obvious to you, may not be —and typically is not —obvious to your reader.

Most especially, don't use phrases like "clearly" or "obviously" in an attempt to bully the reader into accepting something you're having trouble proving. Also, go on the alert whenever you see one of these phrases in someone else's proof.

Finish. At some point in a proof, you'll have established all the essential facts you need. Resist the temptation to quit and leave the reader to draw the "obvious" conclusion. Instead, tie everything together yourself and explain why the original claim follows.

The analogy between good proofs and good programs extends beyond structure. The same rigorous thinking needed for proofs is essential in the design of critical computer systems. When algorithms and protocols only "mostly work" due to reliance on hand-waving arguments, the results can range from problematic to catastrophic. An early example was the Therac 25, a machine that provided radiation therapy to cancer victims, but occasionally killed them with massive overdoses due to a software race condition. A more recent (August 2004) example involved a single faulty command to a computer system used by United and American Airlines that grounded the entire fleet of both companies— and all their passengers!

It is a certainty that we'll all one day be at the mercy of critical computer systems designed by you and your classmates. So we really hope that you'll develop the ability to formulate rock-solid logical arguments that a system actually does what you think it does!

Chapter 1

Propositions

1.1 Mathematical Statements

Definition. A *proposition* is a mathematical statement that is either true or false.

Being true or false doesn't sound like much of a limitation, but it does exclude statements such as, "Wherefore art thou Romeo?" and "Give me an A!".

Being "mathematical" is a more serious restriction. For example, "Albert's wife's name is 'Irene' " is a true statement, and you could prove it by presenting legal documents and the testimony of their children. But it isn't a proposition because it is not a *mathematical* statement. There is no mathematical definition of Albert or Irene, and statements about them are not part of mathematics. Propositions must be about well-defined mathematical objects like numbers, sets, functions, relations, etc., and they must be stated using mathematically precise language. Here are some simple examples.

Proposition 1.1.1. *$2 + 3 = 5$.*

This is a true proposition.

Proposition 1.1.2. *The binary representation of every nonnegative integer starts with a* `1`.

This is a false proposition. It could be fixed by ruling out the nonnegative integer zero. So the following proposition is true:

Proposition 1.1.3. *The binary representation of every positive integer starts with a* `1`.

1.1.1 Problems

Class Problems

Problem 1.1.
Albert announces to his class that he plans to surprise them with a quiz sometime next week.

His students first wonder if the quiz could be on Friday of next next. They reason that it can't: if Albert didn't give the quiz *before* Friday, then by midnight Thursday, they would know the quiz had to be on Friday, and so the quiz wouldn't be a surprise any more.

Next the students wonder whether Albert could give the surprise quiz Thursday. They observe that if the quiz wasn't given *before* Thursday, it would have to be given *on* the Thursday, since they already know it can't be given on Friday. But having figured that out, it wouldn't be a surprise if the quiz was on Thursday either. Similarly, the students reason that the quiz can't be on Wednesday, Tuesday, or Monday. Namely, it's impossible for Albert to give a surprise quiz next week. All the students now relax, having concluded that Albert must have been bluffing.

And since no one expects the quiz, that's why, when Albert gives it on Tuesday next week, it really is a surprise!

What do you think is wrong with the students' reasoning?

1.2 Compound Propositions

It is amazing that people manage to cope with all the ambiguities in the English language. Here are some sentences that illustrate the issue:

1. "You may have cake, or you may have ice cream."

2. "If pigs can fly, then you can understand the Chebyshev bound."

3. "If you can solve any problem we come up with, then you get an A for the course."

4. "Every American has a dream."

What *precisely* do these sentences mean? Can you have both cake and ice cream or must you choose just one dessert? If the second sentence is true, then is the Chebyshev bound incomprehensible? If you can solve some problems we come up with but not all, then do you get an A for the course? And can you still get an A even if you can't solve any of the problems? Does the last sentence imply that all Americans have the same dream or might some of them have different dreams?

Some uncertainty is tolerable in normal conversation. But when we need to formulate ideas precisely —as in mathematics and programming —the ambiguities inherent in everyday language can be a real problem. We can't hope to make an exact argument if we're not sure exactly what the statements mean. So before we start into mathematics, we need to investigate the problem of how to talk about mathematics.

To get around the ambiguity of English, mathematicians have devised a special mini-language for talking about logical relationships. This language mostly uses ordinary English words and phrases such as "or", "implies", and "for all". But mathematicians endow these words with definitions more precise than those found in an ordinary dictionary. Without knowing these definitions, you might sometimes get the gist of statements in this language, but you would regularly get misled about what they really meant.

Surprisingly, in the midst of learning the language of logic, we'll come across the most important open problem in computer science —a problem whose solution could change the world.

1.2.1 Propositions from Propositions

In English, we can modify, combine, and relate propositions with words such as "not", "and", "or", "implies", and "if-then". For example, we can combine three propositions into one like this:

If all humans are mortal **and** all Greeks are human, **then** all Greeks are mortal.

For the next while, we won't be much concerned with the internals of propositions —whether they involve mathematics or Greek mortality —but rather with how propositions are combined and related. So we'll frequently use variables such as P and Q in place of specific propositions such as "All humans are mortal" and "$2 + 3 = 5$". The understanding is that these variables, like propositions, can take on only the values **T** (true) and **F** (false). Such true/false variables are sometimes called *Boolean variables* after their inventor, George —you guessed it —Boole.

NOT, AND, OR

We can precisely define these special words using *truth tables*. For example, if P denotes an arbitrary proposition, then the truth of the proposition "NOT P" is defined by the following truth table:

P	NOT P
T	**F**
F	**T**

The first row of the table indicates that when proposition P is true, the proposition "NOT P" is false. The second line indicates that when P is false, "NOT P" is true. This is probably what you would expect.

In general, a truth table indicates the true/false value of a proposition for each possible setting of the variables. For example, the truth table for the proposition "P AND Q" has four lines, since the two variables can be set in four different ways:

P	Q	P AND Q
T	**T**	**T**
T	**F**	**F**
F	**T**	**F**
F	**F**	**F**

According to this table, the proposition "P AND Q" is true only when P and Q are both true. This is probably the way you think about the word "and."

There is a subtlety in the truth table for "P OR Q":

P	Q	P OR Q
T	**T**	**T**
T	**F**	**T**
F	**T**	**T**
F	**F**	**F**

The third row of this table says that "P OR Q" is true when even if *both* P and Q are true. This isn't always the intended meaning of "or" in everyday speech, but this is the standard definition in mathematical writing. So if a mathematician says, "You may have cake, or you may have ice cream," he means that you *could* have both.

If you want to exclude the possibility of having both having and eating, you should use "exclusive-or" (XOR):

P	Q	P XOR Q
T	**T**	**F**
T	**F**	**T**
F	**T**	**T**
F	**F**	**F**

IMPLIES

The least intuitive connecting word is "implies." Here is its truth table, with the lines labeled so we can refer to them later.

P	Q	P IMPLIES Q	
T	**T**	**T**	(tt)
T	**F**	**F**	(tf)
F	**T**	**T**	(ft)
F	**F**	**T**	(ff)

To experiment with this definition, we're going to examine a famous unsolved problem in mathematics called the Riemann Hypothesis.[1] For now, it doesn't matter what the Riemann Hypothesis happens to be; all that's important in the following example is it is a proposition and no one knows whether it is true or false.

So now let's look at the if-then proposition:

"If the Riemann Hypothesis is true, then $x^2 \geq 0$ for every real number x."

Is this if-then proposition true or false? The answer (that a mathematician would give, anyway) is that it's true! That's because it means that $P \longrightarrow Q$, where the

[1] We'll explain a little about the Riemann Hypothesis in a later chapter (see Definition 4.6.5). Whether or not the Riemann Hypothesis holds is generally considered to be one of the most important unolved problems is mathematics.

if-part, P, called the*hypothesis*, is "the Riemann Hypothesis is true," and the then-part, Q called the *conclusion*, is "$x^2 \geq 0$ for every real number x." Since the conclusion is definitely true, we're on either line (tt) or line (ft) of the truth table. Either way, the proposition as a whole is *true*.

One of our original examples demonstrates an even stranger side of implications.

"If pigs fly, then you can understand the Chebyshev bound."

(Don't take this as an insult to your understanding; we just need to figure out whether this proposition is true or false.) Curiously, the answer has *nothing* to do with whether or not you know anything about the Chebyshev bound. Pigs do not fly, so we're on either line (ft) or line (ff) of the truth table. In both cases, the proposition is *true*!

Sometimes it's worth emphasizing that an implication is true because its hypothesis is false, and we do this by saying it is *vacuously true*.

In contrast, here's an example of a false implication:

"If the moon shines white, then the moon is made of white cheddar."

Yes, the moon shines white. But, no, the moon is not made of white cheddar cheese. So we're on line (tf) of the truth table, and the proposition is false.

The truth table for implications can be summarized in words as follows:

An implication is true exactly when the if-part is false or the then-part is true.

This sentence is worth remembering; a large fraction of all mathematical statements are of the if-then form!

IFF

Mathematicians commonly join propositions in one additional way that doesn't arise in ordinary speech. The proposition "P if and only if Q" asserts that P and Q are logically equivalent; that is, either both are true or both are false.

P	Q	P IFF Q
T	**T**	**T**
T	**F**	**F**
F	**T**	**F**
F	**F**	**T**

The following if-and-only-if statement is true for every real number x:

$$x^2 - 4 \geq 0 \quad \text{iff} \quad |x| \geq 2$$

For some values of x, *both* inequalities are true. For other values of x, *neither* inequality is true . In every case, however, the proposition as a whole is true.

1.2.2 Problems

Class Problems

Problem 1.2.
When the mathematician says to his student, "If a function is not continuous, then it is not differentiable," then letting D stand for "differentiable" and C for continuous, the only proper translation of the mathematician's statement would be

$$\text{NOT}(C) \text{ IMPLIES } \text{NOT}(D),$$

or equivalently,

$$D \text{ IMPLIES } C.$$

But when a mother says to her son, "If you don't do your homework, then you can't watch TV," then letting T stand for "watch TV" and H for "do your homework," a reasonable translation of the mother's statement would be

$$\text{NOT}(H) \text{ IFF } \text{NOT}(T),$$

or equivalently,

$$H \text{ IFF } T.$$

Explain why it is reasonable to translate these two IF-THEN statements in different ways into propositional formulas.

Problem 1.3.
Prove by truth table that OR distributes over AND:

$$[P \text{ OR } (Q \text{ AND } R)] \quad \text{is equivalent to} \quad [(P \text{ OR } Q) \text{ AND } (P \text{ OR } R)] \tag{1.1}$$

Homework Problems

Problem 1.4.
Describe a simple recursive procedure which, given a positive integer argument, n, produces a truth table whose rows are all the assignments of truth values to n propositional variables. For example, for $n = 2$, the table might look like:

T	**T**
T	**F**
F	**T**
F	**F**

Your description can be in English, or a simple program in some familiar language (say Scheme or Java), but if you do write a program, be sure to include some sample output.

1.3 Propositional Logic in Computer Programs

Propositions and logical connectives arise all the time in computer programs. For example, consider the following snippet, which could be either C, C++, or Java:

```
if ( x > 0 || (x <= 0 && y > 100) )
    ⋮
(further instructions)
```

The symbol `||` denotes "or", and the symbol `&&` denotes "and". The *further instructions* are carried out only if the proposition following the word `if` is true. On closer inspection, this big expression is built from two simpler propositions. Let A be the proposition that `x > 0`, and let B be the proposition that `y > 100`. Then we can rewrite the condition this way:

$$A \text{ or } ((\text{not } A) \text{ and } B) \tag{1.2}$$

A truth table reveals that this complicated expression is logically equivalent to

$$A \text{ or } B. \tag{1.3}$$

A	B	A or ((not A) and B)	A or B
T	**T**	**T**	**T**
T	**F**	**T**	**T**
F	**T**	**T**	**T**
F	**F**	**F**	**F**

This means that we can simplify the code snippet without changing the program's behavior:

```
if ( x > 0 || y > 100 )
    ⋮
(further instructions)
```

The equivalence of (1.2) and (1.3) can also be confirmed reasoning by cases:

A is **T**. Then an expression of the form (A or anything) will have truth value **T**. Since both expressions are of this form, both have the same truth value in this case, namely, **T**.

A is **F**. Then (A or P) will have the same truth value as P for any proposition, P. So (1.3) has the same truth value as B. Similarly, (1.2) has the same truth value as ((not **F**) and B), which also has the same value as B. So in this case, both expressions will have the same truth value, namely, the value of B.

Rewriting a logical expression involving many variables in the simplest form is both difficult and important. Simplifying expressions in software might slightly increase the speed of your program. But, more significantly, chip designers face essentially the same challenge. However, instead of minimizing `&&` and `||` symbols in a program, their job is to minimize the number of analogous physical devices on a chip. The payoff is potentially enormous: a chip with fewer devices is smaller, consumes less power, has a lower defect rate, and is cheaper to manufacture.

1.3.1 Cryptic Notation

Programming languages use symbols like && and ! in place of words like "and" and "not". Mathematicians have devised their own cryptic symbols to represent these words, which are summarized in the table below.

English	**Cryptic Notation**
NOT(P)	$\neg P$ (alternatively, $\overline{P}$)
P AND Q	$P \wedge Q$
P OR Q	$P \vee Q$
P IMPLIES Q	$P \longrightarrow Q$
if P then Q	$P \longrightarrow Q$
P IFF Q	$P \longleftrightarrow Q$ (alternatively, $P \quad \text{iff} \quad Q$)

For example, using this notation, "If P and not Q, then R" would be written:

$$(P \wedge \overline{Q}) \longrightarrow R$$

But words such as "OR" and "IMPLIES" generally serve just as well as the cryptic symbols $\wedge$ and $\longrightarrow$, and their meaning is easy to remember. So we'll use the cryptic notation only when it's essential to have a compact formula, and we advise you to do the same.

1.3.2 Logically Equivalent Implications

Do these two sentences say the same thing?

If I am hungry, then I am grumpy.
If I am not grumpy, then I am not hungry.

We can settle the issue by recasting both sentences in terms of propositional logic. Let P be the proposition "I am hungry", and let Q be "I am grumpy". The first sentence says "P implies Q" and the second says "(not Q) implies (not P)". We can compare these two statements in a truth table:

P	Q	P IMPLIES Q	$\overline{Q}$ IMPLIES $\overline{P}$
T	**T**	**T**	**T**
T	**F**	**F**	**F**
F	**T**	**T**	**T**
F	**F**	**T**	**T**

Sure enough, the columns of truth values under these two statements are the same, which precisely means they are equivalent. In general, "(NOT Q) IMPLIES (NOT P)"

is called the *contrapositive* of the implication "P IMPLIES Q." And, as we've just shown, the two are just different ways of saying the same thing.

In contrast, the *converse* of "P IMPLIES Q" is the statement "Q IMPLIES P". In terms of our example, the converse is:

If I am grumpy, then I am hungry.

This sounds like a rather different contention, and a truth table confirms this suspicion:

P	Q	P IMPLIES Q	Q IMPLIES P
T	**T**	**T**	**T**
T	**F**	**F**	**T**
F	**T**	**T**	**F**
F	**F**	**T**	**T**

Thus, an implication *is* logically equivalent to its contrapositive but is *not* equivalent to its converse.

One final relationship: an implication and its converse together are equivalent to an iff statement, specifically, to these two statements together. For example,

If I am grumpy, then I am hungry.
If I am hungry, then I am grumpy.

are equivalent to the single statement:

I am grumpy iff I am hungry.

Once again, we can verify this with a truth table:

P | Q | $(P$ IMPLIES $Q)$ AND $(Q$ IMPLIES $P)$ | Q IFF P
height
T T T T T T T F F F T F F T T F F F F F T T T T

The underlined operators have the same column of truth values, proving that the corresponding formulas are equivalent.

1.3.3 Problems

Class Problems

Problem 1.5.
This problem[2] examines whether the following specifications are *satisfiable*:

1. If the file system is not locked, then
   (a) new messages will be queued.
   (b) new messages will be sent to the messages buffer.

[2]From Rosen, 5th edition, Exercise 1.1.36

(c) the system is functioning normally, and conversely, if the system is functioning normally, then the file system is not locked.

2. If new messages are not queued, then they will be sent to the messages buffer.

3. New messages will not be sent to the message buffer.

(a) Begin by translating the five specifications into propositional formulas using four propositional variables:

$$\begin{aligned} L &::= \text{file system locked,} \\ Q &::= \text{new messages are queued,} \\ B &::= \text{new messages are sent to the message buffer,} \\ N &::= \text{system functioning normally.} \end{aligned}$$

(b) Demonstrate that this set of specifications is satisfiable by describing a single truth assignment for the variables L, Q, B, N and verifying that under this assignment, all the specifications are true.

(c) Argue that the assignment determined in part (b) is the only one that does the job.

Problem 1.6.
Propositional logic comes up in digital circuit design using the convention that **T** corresponds to 1 and **F** to 0. A simple example is a 2-bit *half-adder* circuit. This circuit has 3 binary inputs, a_1, a_0 and b, and 3 binary outputs, c, o_1, o_0. The 2-bit word $a_1 a_0$ gives the binary representation of an integer, k, between 0 and 3. The 3-bit word $c s_1 s_0$ gives the binary representation of $k + b$. The third output bit, c, is called the final *carry bit*.

So if k and b were both 1, then the value of $a_1 a_0$ would be `01` and the value of the output $c s_1 s_0$ would `010`, namely, the 3-bit binary representation of $1 + 1$.

In fact, the final carry bit equals 1 only when all three binary inputs are 1, that is, when $k = 3$ and $b = 1$. In that case, the value of $c s_1 s_0$ is `100`, namely, the binary representation of $3 + 1$.

This 2-bit half-adder could be described by the following formulas:

$$\begin{aligned} c_0 &= b \\ s_0 &= a_0 \ \text{XOR} \ c_0 \\ c_1 &= a_0 \ \text{AND} \ c_0 && \text{the carry into column 1} \\ s_1 &= a_1 \ \text{XOR} \ c_1 \\ c_2 &= a_1 \ \text{AND} \ c_1 && \text{the carry into column 2} \\ c &= c_2. \end{aligned}$$

(a) Generalize the above construction of a 2-bit half-adder to an $n+1$ bit half-adder with inputs $a_n, \ldots, a_1, a_0$ and b for arbitrary $n \geq 0$. That is, give simple formulas for s_i and c_i for $0 \leq i \leq n+1$, where c_i is the carry into column i and $c = c_{n+1}$.

(b) Write similar definitions for the digits and carries in the sum of two $n+1$-bit binary numbers $a_n \ldots a_1 a_0$ and $b_n \ldots b_1 b_0$.

Visualized as digital circuits, the above adders consist of a sequence of single-digit half-adders or adders strung together in series. These circuits mimic ordinary pencil-and-paper addition, where a carry into a column is calculated directly from the carry into the previous column, and the carries have to ripple across all the columns before the carry into the final column is determined. Circuits with this design are called *ripple-carry* adders. Ripple-carry adders are easy to understand and remember and require a nearly minimal number of operations. But the higher-order output bits and the final carry take time proportional to n to reach their final values.

(c) How many of each of the propositional operations does your adder from part (b) use to calculate the sum?

Homework Problems

Problem 1.7.
Considerably faster adder circuits work by computing the values in later columns for both a carry of 0 and a carry of 1, *in parallel*. Then, when the carry from the earlier columns finally arrives, the pre-computed answer can be quickly selected. We'll illustrate this idea by working out the equations for an $n+1$-bit parallel half-adder.

Parallel half-adders are built out of parallel "add1" modules. An $n+1$-bit add1 module takes as input the $n+1$-bit binary representation, $a_n \ldots a_1 a_0$, of an integer, s, and produces as output the binary representation, $c\, p_n \ldots p_1\, p_0$, of $s+1$.

(a) A 1-bit add1 module just has input a_0. Write propositional formulas for its outputs c and p_0.

(b) Explain how to build an $n+1$-bit parallel half-adder from an $n+1$-bit add1 module by writing a propositional formula for the half-adder output, o_i, using only the variables a_i, p_i, and b.

We can build a double-size add1 module with $2(n+1)$ inputs using two single-size add1 modules with $n+1$ inputs. Suppose the inputs of the double-size module are $a_{2n+1}, \ldots, a_1, a_0$ and the outputs are $c, p_{2n+1}, \ldots, p_1, p_0$. The setup is illustrated in Figure 1.2.

Namely, the first single size add1 module handles the first $n+1$ inputs. The inputs to this module are the low-order $n+1$ input bits $a_n, \ldots, a_1, a_0$, and its outputs will serve as the first $n+1$ outputs $p_n, \ldots, p_1, p_0$ of the double-size module. Let $c_{(1)}$ be the remaining carry output from this module.

The inputs to the second single-size module are the higher-order $n+1$ input bits $a_{2n+1}, \ldots, a_{n+2}, a_{n+1}$. Call its first $n+1$ outputs $r_n, \ldots, r_1, r_0$ and let $c_{(2)}$ be its carry.

(c) Write a formula for the carry, c, in terms of $c_{(1)}$ and $c_{(2)}$.

(d) Complete the specification of the double-size module by writing propositional formulas for the remaining outputs, p_i, for $n+1 \leq i \leq 2n+1$. The formula for p_i should only involve the variables a_i, $r_{i-(n+1)}$, and $c_{(1)}$.

(e) Parallel half-adders are exponentially faster than ripple-carry half-adders. Confirm this by determining the largest number of propositional operations required to compute any one output bit of an n-bit add module. (You may assume n is a power of 2.)

1.4 Satisfiability

A proposition is **satisfiable** if some setting of the variables makes the proposition true. For example, P AND $\overline{Q}$ is satisfiable because the expression is true when P is true and Q is false. On the other hand, P AND $\overline{P}$ is not satisfiable because the expression as a whole is false for both settings of P. But determining whether or not a more complicated proposition is satisfiable is not so easy. How about this one?

$$(P \text{ OR } Q \text{ OR } R) \text{ AND } (\overline{P} \text{ OR } \overline{Q}) \text{ AND } (\overline{P} \text{ OR } \overline{R}) \text{ AND } (\overline{R} \text{ OR } \overline{Q})$$

The general problem of deciding whether a proposition is satisfiable is called *SAT*. One approach to SAT is to construct a truth table and check whether or not a **T** ever appears. But this approach is not very efficient; a proposition with n variables has a truth table with 2^n lines, so the effort required to decide about a proposition grows exponentially with the number of variables. For a proposition with just 30 variables, that's already over a billion!

Is there a more *efficient* solution to SAT? In particular, is there some, presumably very ingenious, procedure that determines in a number of steps that grows *polynomially* —like n^2 of n^{14} —instead of exponentially, whether any given proposition is satifiable or not? No one knows. And an awful lot hangs on the answer. An efficient solution to SAT would immediately imply efficient solutions to many, many other important problems involving packing, scheduling, routing, and circuit verification, among other things. This would be wonderful, but there would also be worldwide chaos. Decrypting coded messages would also become an easy task (for most codes). Online financial transactions would be insecure and secret communications could be read by everyone.

Recently there has been exciting progress on *sat-solvers* for practical applications like digital circuit verification. These programs find satisfying assignments with amazing efficiency even for formulas with millions of variables. Unfortunately, it's hard to predict which kind of formulas are amenable to sat-solver meth-

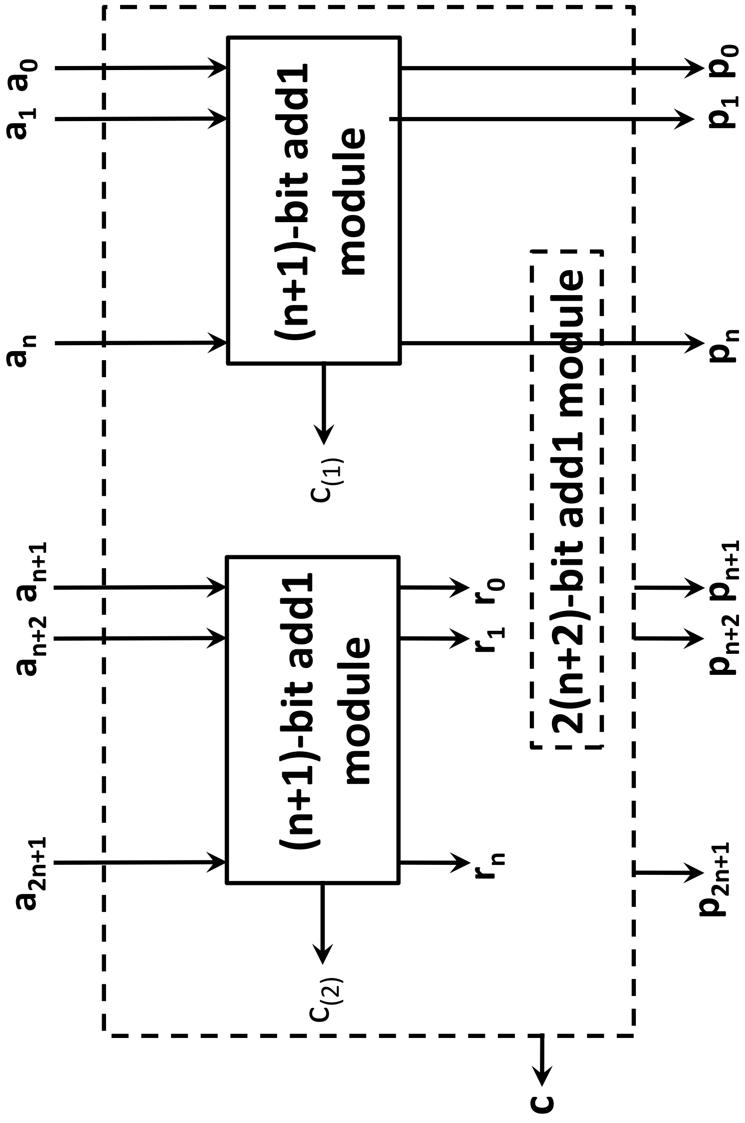


Figure 1.1: Structure of a Double-size Add1 Module.

ods, and for formulas that are NOT satisfiable, sat-solvers generally take exponential time to verify that.

So no one has a good idea how to solve SAT in polynomial time or else to prove that it can't be done —researchers are completely stuck. The problem of determining whether or not SAT has a polynomial time solution is known as the "**P** vs. **NP**" problem. It is the outstanding unanswered question in theoretical computer science. It is also one of the seven Millenium Problems: the Clay Institute will award you $1,000,000 if you solve the **P** vs. **NP** problem.

1.4.1 Problems

Class Problems

Problem 1.8.
This problem[3] examines whether the following specifications are *satisfiable*:

1. If the file system is not locked, then
   (a) new messages will be queued.
   (b) new messages will be sent to the messages buffer.
   (c) the system is functioning normally, and conversely, if the system is functioning normally, then the file system is not locked.
2. If new messages are not queued, then they will be sent to the messages buffer.
3. New messages will not be sent to the message buffer.

(a) Begin by translating the five specifications into propositional formulas using four propositional variables:

$$\begin{aligned} L &::= \text{file system locked,} \\ Q &::= \text{new messages are queued,} \\ B &::= \text{new messages are sent to the message buffer,} \\ N &::= \text{system functioning normally.} \end{aligned}$$

(b) Demonstrate that this set of specifications is satisfiable by describing a single truth assignment for the variables L, Q, B, N and verifying that under this assignment, all the specifications are true.

(c) Argue that the assignment determined in part (b) is the only one that does the job.

[3]From Rosen, 5th edition, Exercise 1.1.36

Problem 1.9.
Propositional logic comes up in digital circuit design using the convention that $\mathbf{T}$ corresponds to 1 and $\mathbf{F}$ to 0. A simple example is a 2-bit *half-adder* circuit. This circuit has 3 binary inputs, a_1, a_0 and b, and 3 binary outputs, c, o_1, o_0. The 2-bit word a_1a_0 gives the binary representation of an integer, k, between 0 and 3. The 3-bit word cs_1s_0 gives the binary representation of $k + b$. The third output bit, c, is called the final *carry bit*.

So if k and b were both 1, then the value of a_1a_0 would be `01` and the value of the output cs_1s_0 would `010`, namely, the 3-bit binary representation of $1 + 1$.

In fact, the final carry bit equals 1 only when all three binary inputs are 1, that is, when $k = 3$ and $b = 1$. In that case, the value of cs_1s_0 is `100`, namely, the binary representation of $3 + 1$.

This 2-bit half-adder could be described by the following formulas:

$$\begin{aligned}
c_0 &= b \\
s_0 &= a_0 \ \text{XOR} \ c_0 \\
c_1 &= a_0 \ \text{AND} \ c_0 && \text{the carry into column 1} \\
s_1 &= a_1 \ \text{XOR} \ c_1 \\
c_2 &= a_1 \ \text{AND} \ c_1 && \text{the carry into column 2} \\
c &= c_2.
\end{aligned}$$

(a) Generalize the above construction of a 2-bit half-adder to an $n + 1$ bit half-adder with inputs $a_n, \ldots, a_1, a_0$ and b for arbitrary $n \geq 0$. That is, give simple formulas for s_i and c_i for $0 \leq i \leq n + 1$, where c_i is the carry into column i and $c = c_{n+1}$.

(b) Write similar definitions for the digits and carries in the sum of two $n + 1$-bit binary numbers $a_n \ldots a_1a_0$ and $b_n \ldots b_1b_0$.

Visualized as digital circuits, the above adders consist of a sequence of single-digit half-adders or adders strung together in series. These circuits mimic ordinary pencil-and-paper addition, where a carry into a column is calculated directly from the carry into the previous column, and the carries have to ripple across all the columns before the carry into the final column is determined. Circuits with this design are called *ripple-carry* adders. Ripple-carry adders are easy to understand and remember and require a nearly minimal number of operations. But the higher-order output bits and the final carry take time proportional to n to reach their final values.

(c) How many of each of the propositional operations does your adder from part (b) use to calculate the sum?

Problem 1.10. (a) A propositional formula is *valid* iff it is equivalent to $\mathbf{T}$. Verify by truth table that

$$(P \text{ IMPLIES } Q) \text{ OR } (Q \text{ IMPLIES } P)$$

is valid.

(b) Let P and Q be propositional formulas. Describe a single propositional formula, R, involving P and Q such that R is valid iff P and Q are equivalent.

(c) A propositional formula is *satisfiable* iff there is an assignment of truth values to its variables —an *environment* —which makes it true. Explain why

P is valid iff $\text{NOT}(P)$ is *not* satisfiable.

(d) A set of propositional formulas $P_1, \ldots, P_k$ is *consistent* iff there is an environment in which they are all true. Write a formula, S, so that the set $P_1, \ldots, P_k$ is *not* consistent iff S is valid.

Problem 1.11.
Considerably faster adder circuits work by computing the values in later columns for both a carry of 0 and a carry of 1, *in parallel*. Then, when the carry from the earlier columns finally arrives, the pre-computed answer can be quickly selected. We'll illustrate this idea by working out the equations for an $n+1$-bit parallel half-adder.

Parallel half-adders are built out of parallel "add1" modules. An $n+1$-bit add1 module takes as input the $n+1$-bit binary representation, $a_n \ldots a_1 a_0$, of an integer, s, and produces as output the binary representation, $c\, p_n \ldots p_1\, p_0$, of $s+1$.

(a) A 1-bit add1 module just has input a_0. Write propositional formulas for its outputs c and p_0.

(b) Explain how to build an $n+1$-bit parallel half-adder from an $n+1$-bit add1 module by writing a propositional formula for the half-adder output, o_i, using only the variables a_i, p_i, and b.

We can build a double-size add1 module with $2(n+1)$ inputs using two single-size add1 modules with $n+1$ inputs. Suppose the inputs of the double-size module are $a_{2n+1}, \ldots, a_1, a_0$ and the outputs are $c, p_{2n+1}, \ldots, p_1, p_0$. The setup is illustrated in Figure 1.2.

Namely, the first single size add1 module handles the first $n+1$ inputs. The inputs to this module are the low-order $n+1$ input bits $a_n, \ldots, a_1, a_0$, and its outputs will serve as the first $n+1$ outputs $p_n, \ldots, p_1, p_0$ of the double-size module. Let $c_{(1)}$ be the remaining carry output from this module.

The inputs to the second single-size module are the higher-order $n+1$ input bits $a_{2n+1}, \ldots, a_{n+2}, a_{n+1}$. Call its first $n+1$ outputs $r_n, \ldots, r_1, r_0$ and let $c_{(2)}$ be its carry.

(c) Write a formula for the carry, c, in terms of $c_{(1)}$ and $c_{(2)}$.

(d) Complete the specification of the double-size module by writing propositional formulas for the remaining outputs, p_i, for $n+1 \leq i \leq 2n+1$. The formula for p_i should only involve the variables a_i, $r_{i-(n+1)}$, and $c_{(1)}$.

(e) Parallel half-adders are exponentially faster than ripple-carry half-adders. Confirm this by determining the largest number of propositional operations required to compute any one output bit of an n-bit add module. (You may assume n is a power of 2.)

1.5 Predicates and Quantifiers

1.5.1 Some More Propositions

A *prime* is an integer greater than one that is not divisible by any integer greater than 1 besides itself, for example, 2, 3, 5, 7, 11, ...

Proposition 1.5.1. *For every nonnegative integer, n, the value of $n^2 + n + 41$ is prime.*

Let's try some numerical experimentation to check this proposition. Let [4]

$$p(n) ::= n^2 + n + 41. \tag{1.4}$$

We begin with $p(0) = 41$ which is prime. $p(1) = 43$ which is prime. $p(2) = 47$ which is prime. $p(3) = 53$ which is prime. $\dots p(20) = 461$ which is prime. Hmmm, starts to look like a plausible claim. In fact we can keep checking through $n = 39$ and confirm that $p(39) = 1601$ is prime.

But $p(40) = 40^2 + 40 + 41 = 41 \cdot 41$, which is not prime. So it's not true that the expression is prime *for all* nonnegative integers. The point is that in general you can't check a claim about an infinite set by checking a finite set of its elements, no matter how large the finite set.

By the way, propositions like this about *all* numbers or other things are so common that there is a special notation for it. With this notation, Proposition 1.5.1 would be

$$\forall n \in \mathbb{N}.\ p(n) \text{ is prime.} \tag{1.5}$$

Here the symbol $\forall$ is read "for all". The symbol $\mathbb{N}$ stands for the set of *nonnegative integers*, namely, 0, 1, 2, 3, ... (ask your instructor for the complete list). The symbol "$\in$" is read as "is a member of," or "belongs to," or simply as "is in". The period after the $\mathbb{N}$ is just a separator between phrases.

Here are two even more extreme examples:

Proposition 1.5.2. *$a^4 + b^4 + c^4 = d^4$ has no solution when a, b, c, d are positive integers.*

Euler (pronounced "oiler") conjectured this in 1769. But the proposition was proven false 218 years later by Noam Elkies at a liberal arts school up Mass Ave. The solution he found was $a = 95800, b = 217519, c = 414560, d = 422481$.

[4]The symbol $::=$ means "equal by definition." It's always ok to simply write "=" instead of $::=$, but reminding the reader that an equality holds by definition can be helpful.

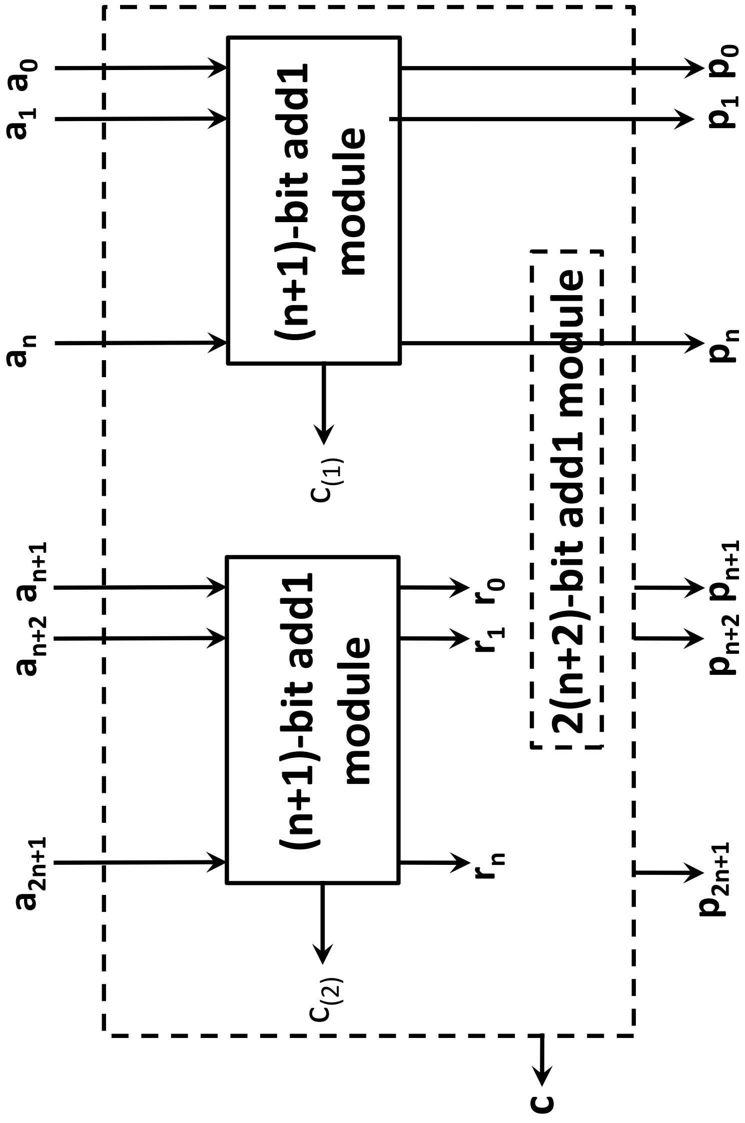


Figure 1.2: Structure of a Double-size Add1 Module.

In logical notation, Proposition 1.5.2 could be written,

$$\forall a \in \mathbb{Z}^+ \, \forall b \in \mathbb{Z}^+ \, \forall c \in \mathbb{Z}^+ \, \forall d \in \mathbb{Z}^+ .\ a^4 + b^4 + c^4 \neq d^4.$$

Here, $\mathbb{Z}^+$ is a symbol for the positive integers. Strings of $\forall$'s like this are usually abbreviated for easier reading:

$$\forall a, b, c, d \in \mathbb{Z}^+ .\ a^4 + b^4 + c^4 \neq d^4.$$

Proposition 1.5.3. *$313(x^3 + y^3) = z^3$ has no solution when $x, y, z \in \mathbb{Z}^+$.*

This proposition is also false, but the smallest counterexample has more than 1000 digits!

Proposition 1.5.4. *Every map can be colored with 4 colors so that adjacent[5] regions have different colors.*

This proposition is true and is known as the "*Four-Color Theorem*". However, there have been many incorrect proofs, including one that stood for 10 years in the late 19th century before the mistake was found. An extremely laborious proof was finally found in 1976 by mathematicians Appel and Haken, who used a complex computer program to categorize the four-colorable maps; the program left a few thousand maps uncategorized, and these were checked by hand by Haken and his assistants—including his 15-year-old daughter. There was a lot of debate about whether this was a legitimate proof: the proof was too big to be checked without a computer, and no one could guarantee that the computer calculated correctly, nor did anyone have the energy to recheck the four-colorings of thousands of maps that were done by hand. Within the past decade a mostly intelligible proof of the Four-Color Theorem was found, though a computer is still needed to check colorability of several hundred special maps.[6]

Proposition 1.5.5 (Goldbach)**.** *Every even integer greater than 2 is the sum of two primes.*

No one knows whether this proposition is true or false. It is known as *Goldbach's Conjecture*, and dates back to 1742.

For a computer scientist, some of the most important things to prove are the "correctness" programs and systems —whether a program or system does what it's supposed to. Programs are notoriously buggy, and there's a growing community of researchers and practitioners trying to find ways to prove program correctness. These efforts have been successful enough in the case of CPU chips that they are now routinely used by leading chip manufacturers to prove chip correctness and avoid mistakes like the notorious Intel division bug in the 1990's.

Developing mathematical methods to verify programs and systems remains an active research area. We'll consider some of these methods later in the course.

[5]Two regions are adjacent only when they share a boundary segment of positive length. They are not considered to be adjacent if their boundaries meet only at a few points.

[6]See `http://www.math.gatech.edu/~thomas/FC/fourcolor.html`

The story of the Four-Color Proof is told in a well-reviewed popular (non-technical) book: "Four Colors Suffice. How the Map Problem was Solved." *Robin Wilson*. Princeton Univ. Press, 2003, 276pp. ISBN 0-691-11533-8.

1.5.2 Predicates

A *predicate* is a proposition whose truth depends on the value of one or more variables. Most of the propositions above were defined in terms of predicates. For example,

$$\text{“}n \text{ is a perfect square”}$$

is a predicate whose truth depends on the value of n. The predicate is true for $n = 4$ since four is a perfect square, but false for $n = 5$ since five is not a perfect square.

Like other propositions, predicates are often named with a letter. Furthermore, a function-like notation is used to denote a predicate supplied with specific variable values. For example, we might name our earlier predicate P:

$$P(n) ::= \text{“}n \text{ is a perfect square”}$$

Now $P(4)$ is true, and $P(5)$ is false.

This notation for predicates is confusingly similar to ordinary function notation. If P is a predicate, then $P(n)$ is either *true* or *false*, depending on the value of n. On the other hand, if p is an ordinary function, like $n^2 + n$, then $p(n)$ is a *numerical quantity*. **Don't confuse these two!**

1.5.3 Quantifiers

There are a couple of assertions commonly made about a predicate: that it is *sometimes* true and that it is *always* true. For example, the predicate

$$\text{“}x^2 \geq 0\text{”}$$

is always true when x is a real number. On the other hand, the predicate

$$\text{“}5x^2 - 7 = 0\text{”}$$

is only sometimes true; specifically, when $x = \pm\sqrt{7/5}$.

There are several ways to express the notions of "always true" and "sometimes true" in English. The table below gives some general formats on the left and specific examples using those formats on the right. You can expect to see such phrases hundreds of times in mathematical writing!

Always True

For all n, $P(n)$ is true.	For all $x \in \mathbb{R}$, $x^2 \geq 0$.
$P(n)$ is true for every n.	$x^2 \geq 0$ for every $x \in \mathbb{R}$.

Sometimes True

There exists an n such that $P(n)$ is true.	There exists an $x \in \mathbb{R}$ such that $5x^2 - 7 = 0$.
$P(n)$ is true for some n.	$5x^2 - 7 = 0$ for some $x \in \mathbb{R}$.
$P(n)$ is true for at least one n.	$5x^2 - 7 = 0$ for at least one $x \in \mathbb{R}$.

All these sentences quantify how often the predicate is true. Specifically, an assertion that a predicate is always true, is called a *universally quantified* statement. An assertion that a predicate is sometimes true, is called an *existentially quantified* statement.

Sometimes English sentences are unclear about quantification:

"If you can solve any problem we come up with, then you get an A for the course."

The phrase "you can solve any problem we can come up with" could reasonably be interpreted as either a universal or existential statement. It might mean:

"You can solve *every* problem we come up with,"

or maybe

"You can solve *at least one* problem we come up with."

It's worth noticing that however the phrase is meant to be quantified, it appears as part of a larger "if . . . , then" statement, which is typical. Quantified statements themselves define predicates or propositions and can be combined with AND, OR, IMPLIES, etc., just like any other statement.

1.5.4 More Cryptic Notation

There are symbols to represent universal and existential quantification, just as there are symbols for "AND" ($\wedge$), "IMPLIES" ($\longrightarrow$), and so forth. In particular, to say that a predicate, $P(x)$, is true for all values of x in some set, D, we write:

$$\forall x \in D.\ P(x) \tag{1.6}$$

The *universal quantifier* symbol $\forall$ is read "for all," so this whole expression (1.6) is read "For all x in D, $P(x)$ is true." Remember that upside-down "A" stands for "**A**ll."

To say that a predicate $P(x)$ is true for at least one value of x in D, we write:

$$\exists x \in D.\ P(x) \tag{1.7}$$

The *existential quantifier* symbol $\exists$, is read "there exists." So expression (1.7) is read, "There exists an x in D such that $P(x)$ is true." Remember that backward "E" stands for "**E**xists."

The symbols $\forall$ and $\exists$ are always followed by a variable —typically with an indication of the set the variable ranges over —and then a predicate, as in the two examples above.

As an example, let Probs be the set of problems we come up with, Solves(x) be the predicate "You can solve problem x", and G be the proposition, "You get an A for the course." Then the two different interpretations of

"If you can solve any problem we come up with, then you get an A for the course."

can be written as follows:

$$(\forall x \in \text{Probs. Solves}(x)) \text{ IMPLIES } G,$$

or maybe

$$(\exists x \in \text{Probs. Solves}(x)) \text{ IMPLIES } G.$$

1.5.5 Mixing Quantifiers

Many mathematical statements involve several quantifiers. For example, *Goldbach's Conjecture* states:

"Every even integer greater than 2 is the sum of two primes."

Let's write this more verbosely to make the use of quantification clearer:

> For every even integer n greater than 2, there exist primes p and q such that $n = p + q$.

Let Evens be the set of even integers greater than 2, and let Primes be the set of primes. Then we can write Goldbach's Conjecture in logic notation as follows:

$$\underbrace{\forall n \in \text{Evens}}_{\substack{\text{for every even} \\ \text{integer } n > 2}} \underbrace{\exists p \in \text{Primes}\ \exists q \in \text{Primes}.}_{\substack{\text{there exist primes} \\ p \text{ and } q \text{ such that}}} n = p + q.$$

1.5.6 Order of Quantifiers

Swapping the order of different kinds of quantifiers (existential or universal) usually changes the meaning of a proposition. For example, let's return to one of our initial, confusing statements:

"Every American has a dream."

This sentence is ambiguous because the order of quantifiers is unclear. Let A be the set of Americans, let D be the set of dreams, and define the predicate $H(a, d)$ to be "American a has dream d.". Now the sentence could mean there is a single dream that every American shares:

$$\exists\, d \in D\ \forall a \in A.\ H(a, d)$$

For example, it might be that every American shares the dream of owning their own home.

Or it could mean that every American has a personal dream:

$$\forall a \in A\ \exists\, d \in D.\ H(a, d)$$

For example, some Americans may dream of a peaceful retirement, while others dream of continuing practicing their profession as long as they live, and still others may dream of being so rich they needn't think at all about work.

Swapping quantifiers in Goldbach's Conjecture creates a patently false statement that every even number ≥ 2 is the sum of *the same* two primes:

$$\underbrace{\exists\, p \in \text{Primes}\ \exists\, q \in \text{Primes}}_{\substack{\text{there exist primes}\\ p \text{ and } q \text{ such that}}}\ \underbrace{\forall n \in \text{Evens}.}_{\substack{\text{for every even}\\ \text{integer } n\, >\, 2}}\ n = p + q.$$

Variables Over One Domain

When all the variables in a formula are understood to take values from the same nonempty set, D, it's conventional to omit mention of D. For example, instead of $\forall x \in D\ \exists y \in D.\ Q(x, y)$ we'd write $\forall x \exists y.\ Q(x, y)$. The unnamed nonempty set that x and y range over is called the *domain of discourse*, or just plain *domain*, of the formula.

It's easy to arrange for all the variables to range over one domain. For example, Goldbach's Conjecture could be expressed with all variables ranging over the domain $\mathbb{N}$ as

$$\forall n.\ n \in \text{Evens IMPLIES } (\exists\, p \exists\, q.\ p \in \text{Primes} \wedge q \in \text{Primes} \wedge n = p + q).$$

1.5.7 Negating Quantifiers

There is a simple relationship between the two kinds of quantifiers. The following two sentences mean the same thing:

It is not the case that everyone likes to snowboard.

There exists someone who does not like to snowboard.

In terms of logic notation, this follows from a general property of predicate formulas:

$$\text{NOT } \forall x.\ P(x) \quad \text{is equivalent to} \quad \exists x.\ \text{NOT } P(x).$$

Similarly, these sentences mean the same thing:

There does not exist anyone who likes skiing over magma.

Everyone dislikes skiing over magma.

We can express the equivalence in logic notation this way:

$$(\text{NOT } \exists x.\ P(x)) \text{ IFF } \forall x.\ \text{NOT } P(x). \tag{1.8}$$

The general principle is that *moving a "not" across a quantifier changes the kind of quantifier.*

1.6 Validity

A propositional formula is called *valid* when it evaluates to **T** no matter what truth values are assigned to the individual propositional variables. For example, the propositional version of the Distributive Law is that P AND $(Q$ OR $R)$ is equivalent to $(P$ AND $Q)$ OR $(P$ AND $R)$. This is the same as saying that

$$[P \text{ AND } (Q \text{ OR } R)] \text{ IFF } [(P \text{ AND } Q) \text{ OR } (P \text{ AND } R)]$$

is valid.

The same idea extends to predicate formulas, but to be valid, a formula now must evaluate to true no matter what values its variables may take over any unspecified domain, and no matter what interpretation a predicate variable may be given. For example, we already observed that the rule for negating a quantifier is captured by the valid assertion (1.8).

Another useful example of a valid assertion is

$$\exists x \forall y.\ P(x,y) \text{ IMPLIES } \forall y \exists x.\ P(x,y). \tag{1.9}$$

Here's an explanation why this is valid:

> Let D be the domain for the variables and P_0 be some binary predicate[7] on D. We need to show that if
>
> $$\exists x \in D\ \forall y \in D.\ P_0(x,y) \tag{1.10}$$
>
> holds under this interpretation, then so does
>
> $$\forall y \in D\ \exists x \in D.\ P_0(x,y). \tag{1.11}$$
>
> So suppose (1.10) is true. Then by definition of $\exists$, this means that some element $d_0 \in D$ has the property that
>
> $$\forall y \in D. P_0(d_0, y).$$
>
> By definition of $\forall$, this means that
>
> $$P_0(d_0, d)$$
>
> is true for all $d \in D$. So given any $d \in D$, there is an element in D, namely, d_0, such that $P_0(d_0, d)$ is true. But that's exactly what (1.11) means, so we've proved that (1.11) holds under this interpretation, as required.

We hope this is helpful as an explanation, but we don't really want to call it a "proof." The problem is that with something as basic as (1.9), it's hard to see what more elementary axioms are ok to use in proving it. What the explanation

[7]That is, a predicate that depends on two variables.

above did was translate the logical formula (1.9) into English and then appeal to the meaning, in English, of "for all" and "there exists" as justification. So this wasn't a proof, just an explanation that once you understand what (1.9) means, it becomes obvious.

In contrast to (1.9), the formula

$$\forall y \exists x.\ P(x, y) \text{ IMPLIES } \exists x \forall y.\ P(x, y). \tag{1.12}$$

is *not* valid. We can prove this just by describing an interpretation where the hypothesis, $\forall y \exists x.\ P(x, y)$, is true but the conclusion, $\exists x \forall y.\ P(x, y)$, is not true. For example, let the domain be the integers and $P(x, y)$ mean $x > y$. Then the hypothesis would be true because, given a value, n, for y we could choose the value of x to be $n + 1$, for example. But under this interpretation the conclusion asserts that there is an integer that is bigger than all integers, which is certainly false. An interpretation like this which falsifies an assertion is called a *counter model* to the assertion.

1.6.1 Problems

Class Problems

Problem 1.12.
A media tycoon has an idea for an all-news television network called LNN: The Logic News Network. Each segment will begin with a definition of the domain of discourse and a few predicates. The day's happenings can then be communicated concisely in logic notation. For example, a broadcast might begin as follows:

> "THIS IS LNN. The domain of discourse is $\{\text{Albert}, \text{Ben}, \text{Claire}, \text{David}, \text{Emily}\}$. Let $D(x)$ be a predicate that is true if x is deceitful. Let $L(x, y)$ be a predicate that is true if x likes y. Let $G(x, y)$ be a predicate that is true if x gave gifts to y."

Translate the following broadcasted logic notation into (English) statements.

(a)

$$(\neg(D(\text{Ben}) \lor D(\text{David}))) \longrightarrow (L(\text{Albert}, \text{Ben}) \land L(\text{Ben}, \text{Albert}))$$

(b)

$$\forall x\ (x = \text{Claire} \land \neg L(x, \text{Emily})) \lor (x \neq \text{Claire} \land L(x, \text{Emily})) \land$$
$$\forall x\ (x = \text{David} \land L(x, \text{Claire})) \lor (x \neq \text{David} \land \neg L(x, \text{Claire}))$$

(c)

$$\neg D(\text{Claire}) \longrightarrow (G(\text{Albert}, \text{Ben}) \land \exists\, x G(\text{Ben}, x))$$

(d)

$$\forall x \exists y \exists z\ (y \neq z) \land L(x, y) \land \neg L(x, z)$$

(e) How could you express "Everyone except for Claire likes Emily" using just propositional connectives *without* using any quantifiers ($\forall, \exists$)? Can you generalize to explain how *any* logical formula over this domain of discourse can be expressed without quantifiers? How big would the formula in the previous part be if it was expressed this way?

Problem 1.13.
The goal of this problem is to translate some assertions about binary strings into logic notation. The domain of discourse is the set of all finite-length binary strings: λ, 0, 1, 00, 01, 10, 11, 000, 001, ... (Here λ denotes the empty string.) In your translations, you may use all the ordinary logic symbols (including =), variables, and the binary symbols `0`, `1` denoting 0, 1.

A string like $\texttt{01}x\texttt{0}y$ of binary symbols and variables denotes the *concatenation* of the symbols and the binary strings represented by the variables. For example, if the value of x is `011` and the value of y is `1111`, then the value of $\texttt{01}x\texttt{0}y$ is the binary string `0101101111`.

Here are some examples of formulas and their English translations. Names for these predicates are listed in the third column so that you can reuse them in your solutions (as we do in the definition of the predicate NO-1S below).

Meaning	Formula	Name
x is a prefix of y	$\exists z\ (xz = y)$	PREFIX(x, y)
x is a substring of y	$\exists u \exists v\ (uxv = y)$	SUBSTRING(x, y)
x is empty or a string of 0's	NOT(SUBSTRING$(\texttt{1}, x)$)	NO-1S(x)

(a) x consists of three copies of some string.

(b) x is an even-length string of 0's.

(c) x does not contain both a 0 and a 1.

(d) x is the binary representation of $2^k + 1$ for some integer $k \geq 0$.

(e) An elegant, slightly trickier way to define NO-1S(x) is:

$$\text{PREFIX}(x, \texttt{0}x). \qquad (*)$$

Explain why (*) is true only when x is a string of 0's.

Problem 1.14.
For each of the logical formulas, indicate whether or not it is true when the domain of discourse is $\mathbb{N}$, (the nonnegative integers 0, 1, 2, ...), $\mathbb{Z}$ (the integers), $\mathbb{Q}$ (the rationals), $\mathbb{R}$ (the real numbers), and $\mathbb{C}$ (the complex numbers). Add a brief explanation to the few cases that merit one.

$$\begin{array}{rrrcl}
 & \exists x & (x^2 & = & 2) \\
\forall x & \exists y & (x^2 & = & y) \\
\forall y & \exists x & (x^2 & = & y) \\
\forall x \neq 0 & \exists y & (xy & = & 1) \\
\exists x & \exists y & (x + 2y & = & 2) \land (2x + 4y = 5)
\end{array}$$

Problem 1.15.
Show that

$$(\forall x \exists y.\ P(x, y)) \longrightarrow \forall z.\ P(z, z)$$

is not valid by describing a counter-model.

Homework Problems

Problem 1.16.
Express each of the following predicates and propositions in formal logic notation. The domain of discourse is the nonnegative integers, $\mathbb{N}$. Moreover, in addition to the propositional operators, variables and quantifiers, you may define predicates using addition, multiplication, and equality symbols, but no *constants* (like 0, 1,. . .) and no *exponentiation* (like x^y). For example, the proposition "n is an even number" could be written

$$\exists m.\ (m + m = n).$$

(a) n is the sum of two fourth-powers (a fourth-power is k^4 for some integer k).

Since the constant 0 is not allowed to appear explicitly, the predicate "$x = 0$" can't be written directly, but note that it could be expressed in a simple way as:

$$x + x = x.$$

Then the predicate $x > y$ could be expressed

$$\exists w.\ (y + w = x) \land (w \neq 0).$$

Note that we've used "$w \neq 0$" in this formula, even though it's technically not allowed. But since "$w \neq 0$" is equivalent to the allowed formula "$\neg(w + w = w)$," we can use "$w \neq 0$" with the understanding that it abbreviates the real thing. And now that we've shown how to express "$x > y$," it's ok to use it too.

(b) $x = 1$.

(c) m is a divisor of n (notation: $m \mid n$)

(d) n is a prime number (hint: use the predicates from the previous parts)

(e) n is a power of 3.

Problem 1.17.
Translate the following sentence into a predicate formula:

> There is a student who has emailed exactly two other people in the class, besides possibly herself.

The domain of discourse should be the set of students in the class; in addition, the only predicates that you may use are

- equality, and
- $E(x, y)$, meaning that "x has sent e-mail to y."

Chapter 2

Patterns of Proof

2.1 The Axiomatic Method

The standard procedure for establishing truth in mathematics was invented by Euclid, a mathematician working in Alexandria, Egypt around 300 BC. His idea was to begin with five *assumptions* about geometry, which seemed undeniable based on direct experience. (For example, "There is a straight line segment between every pair of points.) Propositions like these that are simply accepted as true are called *axioms*.

Starting from these axioms, Euclid established the truth of many additional propositions by providing "proofs". A *proof* is a sequence of logical deductions from axioms and previously-proved statements that concludes with the proposition in question. You probably wrote many proofs in high school geometry class, and you'll see a lot more in this course.

There are several common terms for a proposition that has been proved. The different terms hint at the role of the proposition within a larger body of work.

- Important propositions are called *theorems*.
- A *lemma* is a preliminary proposition useful for proving later propositions.
- A *corollary* is a proposition that follows in just a few logical steps from a theorem.

The definitions are not precise. In fact, sometimes a good lemma turns out to be far more important than the theorem it was originally used to prove.

Euclid's axiom-and-proof approach, now called the *axiomatic method*, is the foundation for mathematics today. In fact, just a handful of axioms, called the axioms Zermelo-Frankel with Choice (*ZFC*), together with a few logical deduction rules, appear to be sufficient to derive essentially all of mathematics. We'll examine these in Chapter 5.

2.1.1 Our Axioms

The ZFC axioms are important in studying and justifying the foundations of mathematics, but for practical purposes, they are much too primitive. Proving theorems in ZFC is a little like writing programs in byte code instead of a full-fledged programming language —by one reckoning, a formal proof in ZFC that $2 + 2 = 4$ requires more than 20,000 steps! So instead of starting with ZFC, we're going to take a *huge* set of axioms as our foundation: we'll accept all familiar facts from high school math!

This will give us a quick launch, but you may find this imprecise specification of the axioms troubling at times. For example, in the midst of a proof, you may find yourself wondering, "Must I prove this little fact or can I take it as an axiom?" Feel free to ask for guidance, but really there is no absolute answer. Just be up front about what you're assuming, and don't try to evade homework and exam problems by declaring everything an axiom!

2.1.2 Logical Deductions

Logical deductions or *inference rules* are used to prove new propositions using previously proved ones.

A fundamental inference rule is *modus ponens*. This rule says that a proof of P together with a proof that P IMPLIES Q is a proof of Q.

Inference rules are sometimes written in a funny notation. For example, *modus ponens* is written:

Rule.

$$\frac{P, \quad P \text{ IMPLIES } Q}{Q}$$

When the statements above the line, called the *antecedents*, are proved, then we can consider the statement below the line, called the *conclusion* or *consequent*, to also be proved.

A key requirement of an inference rule is that it must be *sound*: any assignment of truth values that makes all the antecedents true must also make the consequent true. So if we start off with true axioms and apply sound inference rules, everything we prove will also be true.

There are many other natural, sound inference rules, for example:

Rule.

$$\frac{P \text{ IMPLIES } Q, \quad Q \text{ IMPLIES } R}{P \text{ IMPLIES } R}$$

Rule.

$$\frac{\text{NOT}(P) \text{ IMPLIES NOT}(Q)}{Q \text{ IMPLIES } P}$$

On the other hand,

Rule.

$$\frac{\text{NOT}(P) \text{ IMPLIES NOT}(Q)}{P \text{ IMPLIES } Q}$$

is not sound: if P is assigned **T** and Q is assigned **F**, then the antecedent is true and the consequent is not.

Note that a propositional inference rule is sound precisely when the conjunction (AND) of all its antecedents implies its consequent.

As with axioms, we will not be too formal about the set of legal inference rules. Each step in a proof should be clear and "logical"; in particular, you should state what previously proved facts are used to derive each new conclusion.

2.2 Proof Templates

In principle, a proof can be *any* sequence of logical deductions from axioms and previously proved statements that concludes with the proposition in question. This freedom in constructing a proof can seem overwhelming at first. How do you even *start* a proof?

Here's the good news: many proofs follow one of a handful of standard templates. Each proof has it own details, of course, but these templates at least provide you with an outline to fill in. We'll go through several of these standard patterns, pointing out the basic idea and common pitfalls and giving some examples. Many of these templates fit together; one may give you a top-level outline while others help you at the next level of detail. And we'll show you other, more sophisticated proof techniques later on.

The recipes below are very specific at times, telling you exactly which words to write down on your piece of paper. You're certainly free to say things your own way instead; we're just giving you something you *could* say so that you're never at a complete loss.

2.2.1 Proving an Implication

Propositions of the form "If P, then Q" are called *implications*. This implication is often rephrased as "P IMPLIES Q."

Here are some examples:

- (Quadratic Formula) If $ax^2 + bx + c = 0$ and $a \neq 0$, then
$$x = \left(-b \pm \sqrt{b^2 - 4ac}\right) / 2a.$$
- (Goldbach's Conjecture) If n is an even integer greater than 2, then n is a sum of two primes.
- If $0 \leq x \leq 2$, then $-x^3 + 4x + 1 > 0$.

There are a couple of standard methods for proving an implication.

Method #1

In order to prove that P IMPLIES Q:

1. Write, "Assume P."

2. Show that Q logically follows.

Example.

Theorem 2.2.1. *If* $0 \leq x \leq 2$, *then* $-x^3 + 4x + 1 > 0$.

Before we write a proof of this theorem, we have to do some scratchwork to figure out why it is true.

The inequality certainly holds for $x = 0$; then the left side is equal to 1 and $1 > 0$. As x grows, the $4x$ term (which is positive) initially seems to have greater magnitude than $-x^3$ (which is negative). For example, when $x = 1$, we have $4x = 4$, but $-x^3 = -1$ only. In fact, it looks like $-x^3$ doesn't begin to dominate until $x > 2$. So it seems the $-x^3 + 4x$ part should be nonnegative for all x between 0 and 2, which would imply that $-x^3 + 4x + 1$ is positive.

So far, so good. But we still have to replace all those "seems like" phrases with solid, logical arguments. We can get a better handle on the critical $-x^3 + 4x$ part by factoring it, which is not too hard:

$$-x^3 + 4x = x(2 - x)(2 + x)$$

Aha! For x between 0 and 2, all of the terms on the right side are nonnegative. And a product of nonnegative terms is also nonnegative. Let's organize this blizzard of observations into a clean proof.

Proof. Assume $0 \leq x \leq 2$. Then x, $2 - x$, and $2 + x$ are all nonnegative. Therefore, the product of these terms is also nonnegative. Adding 1 to this product gives a positive number, so:

$$x(2 - x)(2 + x) + 1 > 0$$

Multiplying out on the left side proves that

$$-x^3 + 4x + 1 > 0$$

as claimed. ■

There are a couple points here that apply to all proofs:

- You'll often need to do some scratchwork while you're trying to figure out the logical steps of a proof. Your scratchwork can be as disorganized as you like— full of dead-ends, strange diagrams, obscene words, whatever. But keep your scratchwork separate from your final proof, which should be clear and concise.

- Proofs typically begin with the word "Proof" and end with some sort of doohickey like □ or "q.e.d". The only purpose for these conventions is to clarify where proofs begin and end.

Method #2 - Prove the Contrapositive

An implication ("P IMPLIES Q") is logically equivalent to its *contrapositive*

$$\text{NOT}(Q) \text{ IMPLIES } \text{NOT}(P)$$

Proving one is as good as proving the other, and proving the contrapositive is sometimes easier than proving the original statement. If so, then you can proceed as follows:

1. Write, "We prove the contrapositive:" and then state the contrapositive.
2. Proceed as in Method #1.

Example.

Theorem 2.2.2. *If r is irrational, then $\sqrt{r}$ is also irrational.*

Recall that rational numbers are equal to a ratio of integers and irrational numbers are not. So we must show that if r is *not* a ratio of integers, then $\sqrt{r}$ is also *not* a ratio of integers. That's pretty convoluted! We can eliminate both *not*'s and make the proof straightforward by considering the contrapositive instead.

Proof. We prove the contrapositive: if $\sqrt{r}$ is rational, then r is rational.

Assume that $\sqrt{r}$ is rational. Then there exist integers a and b such that:

$$\sqrt{r} = \frac{a}{b}$$

Squaring both sides gives:

$$r = \frac{a^2}{b^2}$$

Since a^2 and b^2 are integers, r is also rational. ■

2.2.2 Proving an "If and Only If"

Many mathematical theorems assert that two statements are logically equivalent; that is, one holds if and only if the other does. Here is an example that has been known for several thousand years:

> Two triangles have the same side lengths if and only if two side lengths and the angle between those sides are the same.

The phrase "if and only if" comes up so often that it is often abbreviated "iff".

Method #1: Prove Each Statement Implies the Other

The statement "P IFF Q" is equivalent to the two statements "P IMPLIES Q" and "Q IMPLIES P". So you can prove an "iff" by proving *two* implications:

1. Write, "We prove P implies Q and vice-versa."

2. Write, "First, we show P implies Q." Do this by one of the methods in Section 2.2.1.

3. Write, "Now, we show Q implies P." Again, do this by one of the methods in Section 2.2.1.

Method #2: Construct a Chain of Iffs

In order to prove that P is true iff Q is true:

1. Write, "We construct a chain of if-and-only-if implications."

2. Prove P is equivalent to a second statement which is equivalent to a third statement and so forth until you reach Q.

This method sometimes requires more ingenuity than the first, but the result can be a short, elegant proof.

Example. The *standard deviation* of a sequence of values $x_1, x_2, \dots, x_n$ is defined to be:

$$\sqrt{\frac{(x_1-\mu)^2+(x_2-\mu)^2+\cdots+(x_n-\mu)^2}{n}} \tag{2.1}$$

where μ is the *mean* of the values:

$$\mu ::= \frac{x_1+x_2+\cdots+x_n}{n}$$

Theorem 2.2.3. *The standard deviation of a sequence of values $x_1, \dots, x_n$ is zero iff all the values are equal to the mean.*

For example, the standard deviation of test scores is zero if and only if everyone scored exactly the class average.

Proof. We construct a chain of "iff" implications, starting with the statement that the standard deviation (2.1) is zero:

$$\sqrt{\frac{(x_1-\mu)^2+(x_2-\mu)^2+\cdots+(x_n-\mu)^2}{n}} = 0. \tag{2.2}$$

Now since zero is the only number whose square root is zero, equation (2.2) holds iff

$$(x_1-\mu)^2+(x_2-\mu)^2+\cdots+(x_n-\mu)^2 = 0. \tag{2.3}$$

Now squares of real numbers are always nonnegative, so every term on the left hand side of equation (2.3) is nonnegative. This means that (2.3) holds iff

$$\text{Every term on the left hand side of (2.3) is zero.} \tag{2.4}$$

But a term $(x_i-\mu)^2$ is zero iff $x_i = \mu$, so (2.4) is true iff

Every x_i equals the mean.

■

2.2.3 Proof by Cases

Breaking a complicated proof into cases and proving each case separately is a useful, common proof strategy. Here's an amusing example.

Let's agree that given any two people, either they have met or not. If every pair of people in a group has met, we'll call the group a *club*. If every pair of people in a group has not met, we'll call it a group of *strangers*.

Theorem. *Every collection of 6 people includes a club of 3 people or a group of 3 strangers.*

Proof. The proof is by case analysis[1]. Let x denote one of the six people. There are two cases:

1. Among 5 other people besides x, at least 3 have met x.
2. Among the 5 other people, at least 3 have not met x.

Now we have to be sure that at least one of these two cases must hold,[2] but that's easy: we've split the 5 people into two groups, those who have shaken hands with x and those who have not, so one the groups must have at least half the people.

Case 1: Suppose that at least 3 people did meet x.

This case splits into two subcases:

> **Case 1.1:** No pair among those people met each other. Then these people are a group of at least 3 strangers. So the Theorem holds in this subcase.
>
> **Case 1.2:** Some pair among those people have met each other. Then that pair, together with x, form a club of 3 people. So the Theorem holds in this subcase.

This implies that the Theorem holds in Case 1.

Case 2: Suppose that at least 3 people did not meet x.

This case also splits into two subcases:

> **Case 2.1**: Every pair among those people met each other. Then these people are a club of at least 3 people. So the Theorem holds in this subcase.
>
> **Case 2.2:** Some pair among those people have not met each other. Then that pair, together with x, form a group of at least 3 strangers. So the Theorem holds in this subcase.

This implies that the Theorem also holds in Case 2, and therefore holds in all cases.

■

[1]Describing your approach at the outset helps orient the reader.

[2]Part of a case analysis argument is showing that you've covered all the cases. Often this is obvious, because the two cases are of the form "P" and "not P". However, the situation above is not stated quite so simply.

2.2.4 Proof by Contradiction

In a *proof by contradiction* or *indirect proof*, you show that if a proposition were false, then some false fact would be true. Since a false fact can't be true, the proposition had better not be false. That is, the proposition really must be true.

Proof by contradiction is *always* a viable approach. However, as the name suggests, indirect proofs can be a little convoluted. So direct proofs are generally preferable as a matter of clarity.

Method: In order to prove a proposition P by contradiction:

1. Write, "We use proof by contradiction."
2. Write, "Suppose P is false."
3. Deduce something known to be false (a logical contradiction).
4. Write, "This is a contradiction. Therefore, P must be true."

Example

Remember that a number is *rational* if it is equal to a ratio of integers. For example, $3.5 = 7/2$ and $0.1111\cdots = 1/9$ are rational numbers. On the other hand, we'll prove by contradiction that $\sqrt{2}$ is irrational.

Theorem 2.2.4. *$\sqrt{2}$ is irrational.*

Proof. The proof is by contradiction: assume that $\sqrt{2}$ is rational, that is,

$$\sqrt{2} = \frac{n}{d}, \tag{2.5}$$

where n and d are integers. Choose the smallest positive integer denominator, d such that (2.5) holds. We will prove in a moment that the numerator, n, and the denominator, d, are both even. This implies that

$$\frac{n/2}{d/2}$$

is a fraction equal to $\sqrt{2}$ with a smaller positive integer denominator, contradicting the fact that d was smallest such denominator.

Since the assumption that $\sqrt{2}$ is rational leads to this contradiction, the assumption must be false. That is, $\sqrt{2}$ is indeed irrational.

This italicized comment on the implication of the contradiction normally goes without saying, but since this is the first example of a proof by contradiction, we decided to say it.

To prove that n and d have 2 as a common factor, we start by squaring both sides of (2.5) and get $2 = n^2/d^2$, so

$$2d^2 = n^2. \tag{2.6}$$

So 2 is a factor of n^2, which is only possible if 2 is in fact a factor of n.

This means that $n = 2k$ for some integer, k, so

$$n^2 = (2k)^2 = 4k^2. \tag{2.7}$$

Combining (2.6) and (2.7) gives $2d^2 = 4k^2$, so

$$d^2 = 2k^2. \tag{2.8}$$

So 2 is a factor of d^2, which again is only possible if 2 is in fact also a factor of d, as claimed. ■

2.3 *Good* Proofs in Practice

One purpose of a proof is to establish the truth of an assertion with absolute certainty. Mechanically checkable proofs of enormous length or complexity can accomplish this. But humanly intelligible proofs are the only ones that help someone understand the subject. Mathematicians generally agree that important mathematical results can't be fully understood until their proofs are understood. That is why proofs are an important part of the curriculum.

To be understandable and helpful, more is required of a proof than just logical correctness: a good proof must also be clear. Correctness and clarity usually go together; a well-written proof is more likely to be a correct proof, since mistakes are harder to hide.

In practice, the notion of proof is a moving target. Proofs in a professional research journal are generally unintelligible to all but a few experts who know all the terminology and prior results used in the proof. Conversely, proofs in the first weeks of a beginning Math for Computer Science subject would be regarded as tediously long-winded by a professional mathematician. In fact, what might not be acceptable as a good proof in the first weeks of the subject might be perfectly ok in later weeks. But even so, we can offer some general tips on writing good proofs:

State your game plan. A good proof begins by explaining the general line of reasoning, for example, "We use case analysis" or "We argue by contradiction."

Keep a linear flow. Sometimes proofs are written like mathematical mosaics, with juicy tidbits of independent reasoning sprinkled throughout. This is not good. The steps of an argument should follow one another in an intelligible order.

A proof is an essay, not a calculation. Many students initially write proofs the way they compute integrals. The result is a long sequence of expressions without explanation, making it very hard to follow. This is bad. A good proof usually looks like an essay with some equations thrown in. Use complete sentences.

Avoid excessive symbolism. Your reader is probably good at understanding words, but much less skilled at reading arcane mathematical symbols. So use words where you reasonably can.

Revise and simplify. Your readers will be grateful.

Introduce notation thoughtfully. Sometimes an argument can be greatly simplified by introducing a variable, devising a special notation, or defining a new term. But do this sparingly since you're requiring the reader to remember all that new stuff. And remember to actually *define* the meanings of new variables, terms, or notations; don't just start using them!

Structure long proofs. Long programs are usually broken into a hierarchy of smaller procedures. Long proofs are much the same. Facts needed in your proof that are easily stated, but not readily proved are best pulled out and proved in preliminary lemmas. Also, if you are repeating essentially the same argument over and over, try to capture that argument in a general lemma, which you can cite repeatedly instead.

Be wary of the "obvious". When familiar or truly obvious facts are needed in a proof, it's OK to label them as such and to not prove them. But remember that what's obvious to you, may not be —and typically is not —obvious to your reader.

Most especially, don't use phrases like "clearly" or "obviously" in an attempt to bully the reader into accepting something you're having trouble proving. Also, go on the alert whenever you see one of these phrases in someone else's proof.

Finish. At some point in a proof, you'll have established all the essential facts you need. Resist the temptation to quit and leave the reader to draw the "obvious" conclusion. Instead, tie everything together yourself and explain why the original claim follows.

The analogy between good proofs and good programs extends beyond structure. The same rigorous thinking needed for proofs is essential in the design of critical computer systems. When algorithms and protocols only "mostly work" due to reliance on hand-waving arguments, the results can range from problematic to catastrophic. An early example was the Therac 25, a machine that provided radiation therapy to cancer victims, but occasionally killed them with massive overdoses due to a software race condition. A more recent (August 2004) example involved a single faulty command to a computer system used by United and American Airlines that grounded the entire fleet of both companies— and all their passengers!

It is a certainty that we'll all one day be at the mercy of critical computer systems designed by you and your classmates. So we really hope that you'll develop the ability to formulate rock-solid logical arguments that a system actually does what you think it does!

2.3.1 Problems

Class Problems

Problem 2.1.
If we raise an irrational number to an irrational power, can the result be rational? Show that it can by considering $\sqrt{2}^{\sqrt{2}}$ and arguing by cases.

Problem 2.2.
Identify exactly where the bugs are in each of the following bogus proofs.[3]

(a) **Bogus Claim**: $1/8 > 1/4$.

Bogus proof.

$$\begin{aligned} 3 &> 2 \\ 3\log_{10}(1/2) &> 2\log_{10}(1/2) \\ \log_{10}(1/2)^3 &> \log_{10}(1/2)^2 \\ (1/2)^3 &> (1/2)^2, \end{aligned}$$

and the claim now follows by the rules for multiplying fractions. ■

(b) *Bogus proof*: $1¢ = \$0.01 = (\$0.1)^2 = (10¢)^2 = 100¢ = \$1$. ■

(c) **Bogus Claim**: If a and b are two equal real numbers, then $a = 0$.

Bogus proof.

$$\begin{aligned} a &= b \\ a^2 &= ab \\ a^2 - b^2 &= ab - b^2 \\ (a-b)(a+b) &= (a-b)b \\ a+b &= b \\ a &= 0. \end{aligned}$$

■

Problem 2.3.
It's a fact that the Arithmetic Mean is at least as large the Geometric Mean, namely,

$$\frac{a+b}{2} \geq \sqrt{ab}$$

[3] From Stueben, Michael and Diane Sandford. *Twenty Years Before the Blackboard*, Mathematical Association of America, ©1998.

for all nonnegative real numbers a and b. But there's something objectionable about the following proof of this fact. What's the objection, and how would you fix it?

Bogus proof.

$$\begin{aligned} \frac{a+b}{2} &\overset{?}{\geq} \sqrt{ab}, && \text{so} \\ a+b &\overset{?}{\geq} 2\sqrt{ab}, && \text{so} \\ a^2+2ab+b^2 &\overset{?}{\geq} 4ab, && \text{so} \\ a^2-2ab+b^2 &\overset{?}{\geq} 0, && \text{so} \\ (a-b)^2 &\geq 0 && \text{which we know is true.} \end{aligned}$$

The last statement is true because $a-b$ is a real number, and the square of a real number is never negative. This proves the claim. ■

Problem 2.4.
Generalize the proof of Theorem 2.2.4 that $\sqrt{2}$ is irrational. For example, how about $\sqrt[3]{2}$?

Problem 2.5.
Here is a different proof that $\sqrt{2}$ is irrational, taken from the American Mathematical Monthly, v.116, #1, Jan. 2009, p.69:

Proof. Suppose for the sake of contradiction that $\sqrt{2}$ is rational, and choose the least integer, $q > 0$, such that $\left(\sqrt{2}-1\right)q$ is a nonnegative integer. Let $q' ::= \left(\sqrt{2}-1\right)q$. Clearly $0 < q' < q$. But an easy computation shows that $\left(\sqrt{2}-1\right)q'$ is a nonnegative integer, contradicting the minimality of q. ■

(a) This proof was written for an audience of college teachers, and at this point it is a little more concise than desirable. Write out a more complete version which includes an explanation of each step.

(b) Now that you have justified the steps in this proof, do you have a preference for one of these proofs over the other? Why? Discuss these questions with your teammates for a few minutes and summarize your team's answers on your whiteboard.

Problem 2.6.
Here is a generalization of Problem 2.4 that you may not have thought of:

Lemma 2.3.1. *Let the coefficients of the polynomial $a_0+a_1x+a_2x^2+\cdots+a_{n-1}x^{m-1}+x^m$ be integers. Then any real root of the polynomial is either integral or irrational.*

(a) Explain why Lemma 2.3.1 immediately implies that $\sqrt[m]{k}$ is irrational whenever k is not an mth power of some integer.

(b) Collaborate with your tablemates to write a clear, textbook quality proof of Lemma 2.3.1 on your whiteboard. (Besides clarity and correctness, textbook quality requires good English with proper punctuation. When a real textbook writer does this, it usually takes multiple revisions; if you're satisfied with your first draft, you're probably misjudging.) You may find it helpful to appeal to the following:
Lemma 2.3.2. *If a prime, p, is a factor of some power of an integer, then it is a factor of that integer.*

You may assume Lemma 2.3.2 without writing down its proof, but see if you can explain why it is true.

Homework Problems

Problem 2.7.
Show that $\log_7 n$ is either an integer or irrational, where n is a positive integer. Use whatever familiar facts about integers and primes you need, but explicitly state such facts.

Problem 2.8.
For $n = 40$, the value of polynomial $p(n) ::= n^2 + n + 41$ 1.4, is not prime, as noted in Section 1.5. But we could have predicted based on general principles that no nonconstant polynomial, $q(n)$, with integer coefficients can map each nonnegative integer into a prime number. Prove it.

Hint: Let $c ::= q(0)$ be the constant term of q. Consider two cases: c is not prime, and c is prime. In the second case, note that $q(cn)$ is a multiple of c for all $n \in \mathbb{Z}$. You may assume the familiar fact that the magnitude (absolute value) of any nonconstant polynomial, $q(n)$, grows unboundedly as n grows.

Problem 2.9.
The fact that that there are irrational numbers a, b such that a^b is rational was proved in Problem 2.1. Unfortunately, that proof was *nonconstructive*: it didn't reveal a specific pair, a, b, with this property. But in fact, it's easy to do this: let $a ::= \sqrt{2}$ and $b ::= 2\log_2 3$.

We know $\sqrt{2}$ is irrational, and obviously $a^b = 3$. Finish the proof that this a, b pair works, by showing that $2\log_2 3$ is irrational.

Chapter 3

Induction

3.1 The Well Ordering Principle

Every *nonempty* set of *nonnegative integers* has a *smallest* element.

This statement is known as The *Well Ordering Principle*. Do you believe it? Seems sort of obvious, right? But notice how tight it is: it requires a *nonempty* set —it's false for the empty set which has *no* smallest element because it has no elements at all! And it requires a set of *nonnegative* integers —it's false for the set of *negative* integers and also false for some sets of nonnegative *rationals* —for example, the set of positive rationals. So, the Well Ordering Principle captures something special about the nonnegative integers.

3.1.1 Well Ordering Proofs

While the Well Ordering Principle may seem obvious, it's hard to see offhand why it is useful. But in fact, it provides one of the most important proof rules in discrete mathematics.

In fact, looking back, we took the Well Ordering Principle for granted in proving that $\sqrt{2}$ is irrational. That proof assumed that for any positive integers m and n, the fraction m/n can be written in *lowest terms*, that is, in the form m'/n' where m' and n' are positive integers with no common factors. How do we know this is always possible?

Suppose to the contrary that there were $m, n \in \mathbb{Z}^+$ such that the fraction m/n cannot be written in lowest terms. Now let C be the set of positive integers that are numerators of such fractions. Then $m \in C$, so C is nonempty. Therefore, by Well Ordering, there must be a smallest integer, $m_0 \in C$. So by definition of C, there is an integer $n_0 > 0$ such that

$$\text{the fraction } \frac{m_0}{n_0} \text{ cannot be written in lowest terms.}$$

This means that m_0 and n_0 must have a common factor, $p > 1$. But

$$\frac{m_0/p}{n_0/p} = \frac{m_0}{n_0},$$

so any way of expressing the left hand fraction in lowest terms would also work for m_0/n_0, which implies

$$\text{the fraction } \frac{m_0/p}{n_0/p} \text{ cannot be in written in lowest terms either.}$$

So by definition of C, the numerator, m_0/p, is in C. But $m_0/p < m_0$, which contradicts the fact that m_0 is the smallest element of C.

Since the assumption that C is nonempty leads to a contradiction, it follows that C must be empty. That is, that there are no numerators of fractions that can't be written in lowest terms, and hence there are no such fractions at all.

We've been using the Well Ordering Principle on the sly from early on!

3.1.2 Template for Well Ordering Proofs

More generally, there is a standard way to use Well Ordering to prove that some property, $P(n)$ holds for every nonnegative integer, n. Here is a standard way to organize such a well ordering proof:

To prove that "$P(n)$ is true for all $n \in \mathbb{N}$" using the Well Ordering Principle:

- Define the set, C, of *counterexamples* to P being true. Namely, define[a]

$$C ::= \{n \in \mathbb{N} \mid P(n) \text{ is false}\}.$$

- Assume for proof by contradiction that C is nonempty.
- By the Well Ordering Principle, there will be a smallest element, n, in C.
- Reach a contradiction (somehow) —often by showing how to use n to find another member of C that is smaller than n. (This is the open-ended part of the proof task.)
- Conclude that C must be empty, that is, no counterexamples exist. QED

[a]The notation $\{n \mid P(n)\}$ means "the set of all elements n, for which $P(n)$ is true.

3.1.3 Summing the Integers

Let's use this this template to prove

Theorem.

$$1 + 2 + 3 + \cdots + n = n(n+1)/2 \tag{3.1}$$

for all nonnegative integers, n.

First, we better address of a couple of ambiguous special cases before they trip us up:

- If $n = 1$, then there is only one term in the summation, and so $1+2+3+\cdots+n$ is just the term 1. Don't be misled by the appearance of 2 and 3 and the suggestion that 1 and n are distinct terms!
- If $n \leq 0$, then there are no terms at all in the summation. By convention, the sum in this case is 0.

So while the dots notation is convenient, you have to watch out for these special cases where the notation is misleading! (In fact, whenever you see the dots, you should be on the lookout to be sure you understand the pattern, watching out for the beginning and the end.)

We could have eliminated the need for guessing by rewriting the left side of (3.1) with *summation notation*:

$$\sum_{i=1}^{n} i \qquad \text{or} \qquad \sum_{1 \leq i \leq n} i.$$

Both of these expressions denote the sum of all values taken by the expression to the right of the sigma as the variable, i, ranges from 1 to n. Both expressions make it clear what (3.1) means when $n = 1$. The second expression makes it clear that when $n = 0$, there are no terms in the sum, though you still have to know the convention that a sum of no numbers equals 0 (the *product* of no numbers is 1, by the way).

OK, back to the proof:

Proof. By contradiction. Assume that the theorem is *false*. Then, some nonnegative integers serve as *counterexamples* to it. Let's collect them in a set:

$$C ::= \left\{ n \in \mathbb{N} \mid 1 + 2 + 3 + \cdots + n \neq \frac{n(n+1)}{2} \right\}.$$

By our assumption that the theorem admits counterexamples, C is a nonempty set of nonnegative integers. So, by the Well Ordering Principle, C has a minimum element, call it c. That is, c is the *smallest counterexample* to the theorem.

Since c is the smallest counterexample, we know that (3.1) is false for $n = c$ but true for all nonnegative integers $n < c$. But (3.1) is true for $n = 0$, so $c > 0$. This means $c-1$ is a nonnegative integer, and since it is less than c, equation (3.1) is true for $c-1$. That is,

$$1+2+3+\cdots+(c-1)=\frac{(c-1)c}{2}.$$

But then, adding c to both sides we get

$$1+2+3+\cdots+(c-1)+c=\frac{(c-1)c}{2}+c=\frac{c^2-c+2c}{2}=\frac{c(c+1)}{2},$$

which means that (3.1) does hold for c, after all! This is a contradiction, and we are done. ■

3.1.4 Factoring into Primes

We've previously taken for granted the *Prime Factorization Theorem* that every integer greater than one has a unique[1] expression as a product of prime numbers. This is another of those familiar mathematical facts which are not really obvious. We'll prove the uniqueness of prime factorization in a later chapter, but well ordering gives an easy proof that every integer greater than one can be expressed as *some* product of primes.

Theorem 3.1.1. *Every natural number can be factored as a product of primes.*

Proof. The proof is by Well Ordering.

Let C be the set of all integers greater than one that cannot be factored as a product of primes. We assume C is not empty and derive a contradiction.

If C is not empty, there is a least element, $n \in C$, by Well Ordering. The n can't be prime, because a prime by itself is considered a (length one) product of primes and no such products are in C.

So n must be a product of two integers a and b where $1 < a, b < n$. Since a and b are smaller than the smallest element in C, we know that $a, b \notin C$. In other words, a can be written as a product of primes $p_1p_2\cdots p_k$ and b as a product of primes $q_1\cdots q_l$. Therefore, $n = p_1\cdots p_kq_1\cdots q_l$ can be written as a product of primes, contradicting the claim that $n \in C$. Our assumption that $C \neq \emptyset$ must therefore be false. ■

3.1.5 Problems

Practice Problems

Problem 3.1.

For practice using the Well Ordering Principle, fill in the template of an easy to prove fact: every amount of postage that can be assembled using only 10 cent and 15 cent stamps is divisible by 5.

[1]. . . unique up to the order in which the prime factors appear

In particular, Let $S(n)$ mean that exactly n cents postage can be assembled using only 10 and 15 cent stamps. Then the proof shows that

$$S(n) \text{ IMPLIES } 5 \mid n, \quad \text{for all nonnegative integers } n. \tag{*}$$

Fill in the missing portions (indicated by "...") of the following proof of (*).

> Let C be the set of *counterexamples* to (*), namely
>
> $$C ::= \{n \mid \ldots\}$$
>
> Assume for the purpose of obtaining a contradiction that C is nonempty. Then by the WOP, there is a smallest number, $m \in C$. This m must be positive because ....
>
> But if $S(m)$ holds and m is positive, then $S(m-10)$ or $S(m-15)$ must hold, because ....
>
> So suppose $S(m-10)$ holds. Then $5 \mid (m-10)$, because...
>
> But if $5 \mid (m-10)$, then obviously $5 \mid m$, contradicting the fact that m is a counterexample.
>
> Next, if $S(m-15)$ holds, we arrive at a contradiction in the same way.
>
> Since we get a contradiction in both cases, we conclude that...
>
> which proves that (*) holds.

Class Problems

Problem 3.2.
The proof below uses the Well Ordering Principle to prove that every amount of postage that can be assembled using only 6 cent and 15 cent stamps, is divisible by 3. Let the notation "$j \mid k$" indicate that integer j is a divisor of integer k, and let $S(n)$ mean that exactly n cents postage can be assembled using only 6 and 15 cent stamps. Then the proof shows that

$$S(n) \text{ IMPLIES } 3 \mid n, \quad \text{for all nonnegative integers } n. \tag{*}$$

Fill in the missing portions (indicated by "...") of the following proof of (*).

> Let C be the set of *counterexamples* to (*), namely[2]
>
> $$C ::= \{n \mid \ldots\}$$
>
> Assume for the purpose of obtaining a contradiction that C is nonempty. Then by the WOP, there is a smallest number, $m \in C$. This m must be positive because....

[2]The notation "$\{n \mid \ldots\}$" means "the set of elements, n, such that ...."

But if $S(m)$ holds and m is positive, then $S(m-6)$ or $S(m-15)$ must hold, because....

So suppose $S(m-6)$ holds. Then $3 \mid (m-6)$, because...

But if $3 \mid (m-6)$, then obviously $3 \mid m$, contradicting the fact that m is a counterexample.

Next, if $S(m-15)$ holds, we arrive at a contradiction in the same way. Since we get a contradiction in both cases, we conclude that...

which proves that (*) holds.

Problem 3.3.
Euler's Conjecture in 1769 was that there are no positive integer solutions to the equation

$$a^4 + b^4 + c^4 = d^4.$$

Integer values for a, b, c, d that do satisfy this equation, were first discovered in 1986. So Euler guessed wrong, but it took more two hundred years to prove it.

Now let's consider Lehman's equation, similar to Euler's but with some coefficients:

$$8a^4 + 4b^4 + 2c^4 = d^4 \tag{3.2}$$

Prove that Lehman's equation (3.2) really does not have any positive integer solutions.

Hint: Consider the minimum value of a among all possible solutions to (3.2).

Problem 3.4.
Use the Well Ordering Principle to prove that

$$\sum_{k=0}^{n} k^2 = \frac{n(n+1)(2n+1)}{6}. \tag{3.3}$$

for all nonnegative integers, n.

Homework Problems

Problem 3.5.
Use the Well Ordering Principle to prove that any integer greater than or equal to 8 can be represented as the sum of integer multiples of 3 and 5.

3.2 Induction

Induction is by far the most powerful and commonly-used proof technique in discrete mathematics and computer science. In fact, the use of induction is a defining characteristic of *discrete* —as opposed to *continuous* —mathematics. To understand how it works, suppose there is a professor who brings to class a bottomless bag of assorted miniature candy bars. She offers to share the candy in the following way. First, she lines the students up in order. Next she states two rules:

1. The student at the beginning of the line gets a candy bar.

2. If a student gets a candy bar, then the following student in line also gets a candy bar.

Let's number the students by their order in line, starting the count with 0, as usual in Computer Science. Now we can understand the second rule as a short description of a whole sequence of statements:

- If student 0 gets a candy bar, then student 1 also gets one.

- If student 1 gets a candy bar, then student 2 also gets one.

- If student 2 gets a candy bar, then student 3 also gets one.

$$\vdots$$

Of course this sequence has a more concise mathematical description:

> If student n gets a candy bar, then student $n+1$ gets a candy bar, for all nonnegative integers n.

So suppose you are student 17. By these rules, are you entitled to a miniature candy bar? Well, student 0 gets a candy bar by the first rule. Therefore, by the second rule, student 1 also gets one, which means student 2 gets one, which means student 3 gets one as well, and so on. By 17 applications of the professor's second rule, you get your candy bar! Of course the rules actually guarantee a candy bar to *every* student, no matter how far back in line they may be.

3.2.1 Ordinary Induction

The reasoning that led us to conclude every student gets a candy bar is essentially all there is to induction.

The Principle of Induction.

Let $P(n)$ be a predicate. If

- $P(0)$ is true, and
- $P(n)$ IMPLIES $P(n+1)$ for all nonnegative integers, n,

then

- $P(m)$ is true for all nonnegative integers, m.

Since we're going to consider several useful variants of induction in later sections, we'll refer to the induction method described above as *ordinary induction* when we need to distinguish it. Formulated as a proof rule, this would be

Rule. ***Induction Rule***

$$\frac{P(0), \quad \forall n \in \mathbb{N}\,[P(n) \text{ IMPLIES } P(n+1)]}{\forall m \in \mathbb{N}.\, P(m)}$$

This general induction rule works for the same intuitive reason that all the students get candy bars, and we hope the explanation using candy bars makes it clear why the soundness of the ordinary induction can be taken for granted. In fact, the rule is so obvious that it's hard to see what more basic principle could be used to justify it.[3] What's not so obvious is how much mileage we get by using it.

Using Ordinary Induction

Ordinary induction often works directly in proving that some statement about nonnegative integers holds for all of them. For example, here is the formula for the sum of the nonnegative integer that we already proved (equation (3.1)) using the Well Ordering Principle:

Theorem 3.2.1. *For all $n \in \mathbb{N}$,*

$$1+2+3+\cdots+n = \frac{n(n+1)}{2} \tag{3.4}$$

This time, let's use the Induction Principle to prove Theorem 3.2.1.

Suppose that we define predicate $P(n)$ to be the equation (3.4). Recast in terms of this predicate, the theorem claims that $P(n)$ is true for all $n \in \mathbb{N}$. This is great, because the induction principle lets us reach precisely that conclusion, provided we establish two simpler facts:

[3] But see section 3.3.

- $P(0)$ is true.
- For all $n \in \mathbb{N}$, $P(n)$ IMPLIES $P(n+1)$.

So now our job is reduced to proving these two statements. The first is true because $P(0)$ asserts that a sum of zero terms is equal to $0(0+1)/2 = 0$, which is true by definition. The second statement is more complicated. But remember the basic plan for proving the validity of any implication: *assume* the statement on the left and then *prove* the statement on the right. In this case, we assume $P(n)$ in order to prove $P(n+1)$, which is the equation

$$1 + 2 + 3 + \cdots + n + (n+1) = \frac{(n+1)(n+2)}{2}. \tag{3.5}$$

These two equations are quite similar; in fact, adding $(n+1)$ to both sides of equation (3.4) and simplifying the right side gives the equation (3.5):

$$\begin{aligned} 1 + 2 + 3 + \cdots + n + (n+1) &= \frac{n(n+1)}{2} + (n+1) \\ &= \frac{(n+2)(n+1)}{2} \end{aligned}$$

Thus, if $P(n)$ is true, then so is $P(n+1)$. This argument is valid for every nonnegative integer n, so this establishes the second fact required by the induction principle. Therefore, the induction principle says that the predicate $P(m)$ is true for all nonnegative integers, m, so the theorem is proved.

A Template for Induction Proofs

The proof of Theorem 3.2.1 was relatively simple, but even the most complicated induction proof follows exactly the same template. There are five components:

1. **State that the proof uses induction.** This immediately conveys the overall structure of the proof, which helps the reader understand your argument.

2. **Define an appropriate predicate $P(n)$.** The eventual conclusion of the induction argument will be that $P(n)$ is true for all nonnegative n. Thus, you should define the predicate $P(n)$ so that your theorem is equivalent to (or follows from) this conclusion. Often the predicate can be lifted straight from the claim, as in the example above. The predicate $P(n)$ is called the *induction hypothesis*. Sometimes the induction hypothesis will involve several variables, in which case you should indicate which variable serves as n.

3. **Prove that $P(0)$ is true.** This is usually easy, as in the example above. This part of the proof is called the *base case* or *basis step*.

4. **Prove that $P(n)$ implies $P(n+1)$ for every nonnegative integer n.** This is called the *inductive step*. The basic plan is always the same: assume that $P(n)$

is true and then use this assumption to prove that $P(n+1)$ is true. These two statements should be fairly similar, but bridging the gap may require some ingenuity. Whatever argument you give must be valid for every nonnegative integer n, since the goal is to prove the implications $P(0) \rightarrow P(1)$, $P(1) \rightarrow P(2)$, $P(2) \rightarrow P(3)$, etc. all at once.

5. **Invoke induction.** Given these facts, the induction principle allows you to conclude that $P(n)$ is true for all nonnegative n. This is the logical capstone to the whole argument, but it is so standard that it's usual not to mention it explicitly,

Explicitly labeling the *base case* and *inductive step* may make your proofs clearer.

A Clean Writeup

The proof of Theorem 3.2.1 given above is perfectly valid; however, it contains a lot of extraneous explanation that you won't usually see in induction proofs. The writeup below is closer to what you might see in print and should be prepared to produce yourself.

Proof. We use induction. The induction hypothesis, $P(n)$, will be equation (3.4).

Base case: $P(0)$ is true, because both sides of equation (3.4) equal zero when $n = 0$.

Inductive step: Assume that $P(n)$ is true, where n is any nonnegative integer. Then

$$\begin{aligned} 1+2+3+\cdots+n+(n+1) &= \frac{n(n+1)}{2}+(n+1) && \text{(by induction hypothesis)} \\ &= \frac{(n+1)(n+2)}{2} && \text{(by simple algebra)} \end{aligned}$$

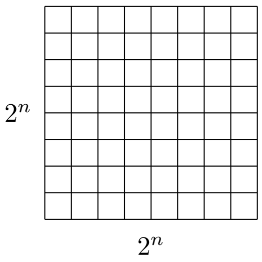
which proves $P(n+1)$.

So it follows by induction that $P(n)$ is true for all nonnegative n. ■

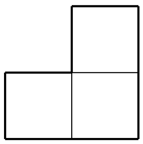
Induction was helpful for *proving the correctness* of this summation formula, but not helpful for *discovering* it in the first place. Tricks and methods for finding such formulas will appear in a later chapter.

Courtyard Tiling

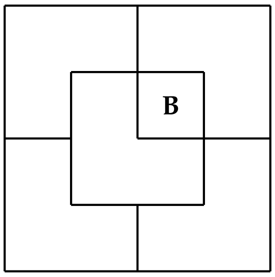
During the development of MIT's famous Stata Center, costs rose further and further over budget, and there were some radical fundraising ideas. One rumored plan was to install a big courtyard with dimensions $2^n \times 2^n$:



One of the central squares would be occupied by a statue of a wealthy potential donor. Let's call him "Bill". (In the special case $n = 0$, the whole courtyard consists of a single central square; otherwise, there are four central squares.) A complication was that the building's unconventional architect, Frank Gehry, was alleged to require that only special L-shaped tiles be used:



A courtyard meeting these constraints exists, at least for $n = 2$:



For larger values of n, is there a way to tile a $2^n \times 2^n$ courtyard with L-shaped tiles and a statue in the center? Let's try to prove that this is so.

Theorem 3.2.2. *For all $n \geq 0$ there exists a tiling of a $2^n \times 2^n$ courtyard with Bill in a central square.*

Proof. (doomed attempt) The proof is by induction. Let $P(n)$ be the proposition that there exists a tiling of a $2^n \times 2^n$ courtyard with Bill in the center.

Base case: $P(0)$ is true because Bill fills the whole courtyard.

Inductive step: Assume that there is a tiling of a $2^n \times 2^n$ courtyard with Bill in the center for some $n \geq 0$. We must prove that there is a way to tile a $2^{n+1} \times 2^{n+1}$ courtyard with Bill in the center .... ■

Now we're in trouble! The ability to tile a smaller courtyard with Bill in the center isn't much help in tiling a larger courtyard with Bill in the center. We haven't figured out how to bridge the gap between $P(n)$ and $P(n + 1)$.

So if we're going to prove Theorem 3.2.2 by induction, we're going to need some *other* induction hypothesis than simply the statement about n that we're trying to prove.

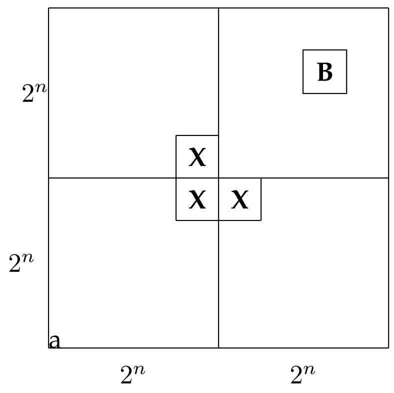
When this happens, your first fallback should be to look for a *stronger* induction hypothesis; that is, one which implies your previous hypothesis. For example, we could make $P(n)$ the proposition that for *every* location of Bill in a $2^n \times 2^n$ courtyard, there exists a tiling of the remainder.

This advice may sound bizarre: "If you can't prove something, try to prove something grander!" But for induction arguments, this makes sense. In the inductive step, where you have to prove $P(n)$ IMPLIES $P(n+1)$, you're in better shape because you can *assume* $P(n)$, which is now a more powerful statement. Let's see how this plays out in the case of courtyard tiling.

Proof. (successful attempt) The proof is by induction. Let $P(n)$ be the proposition that for every location of Bill in a $2^n \times 2^n$ courtyard, there exists a tiling of the remainder.

Base case: $P(0)$ is true because Bill fills the whole courtyard.

Inductive step: Assume that $P(n)$ is true for some $n \geq 0$; that is, for every location of Bill in a $2^n \times 2^n$ courtyard, there exists a tiling of the remainder. Divide the $2^{n+1} \times 2^{n+1}$ courtyard into four quadrants, each $2^n \times 2^n$. One quadrant contains Bill (**B** in the diagram below). Place a temporary Bill (**X** in the diagram) in each of the three central squares lying outside this quadrant:



Now we can tile each of the four quadrants by the induction assumption. Replacing the three temporary Bills with a single L-shaped tile completes the job. This proves that $P(n)$ implies $P(n+1)$ for all $n \geq 0$. The theorem follows as a special case. ■

This proof has two nice properties. First, not only does the argument guarantee that a tiling exists, but also it gives an algorithm for finding such a tiling. Second, we have a stronger result: if Bill wanted a statue on the edge of the courtyard, away from the pigeons, we could accommodate him!

Strengthening the induction hypothesis is often a good move when an induction proof won't go through. But keep in mind that the stronger assertion must actually be *true*; otherwise, there isn't much hope of constructing a valid proof! Sometimes finding just the right induction hypothesis requires trial, error, and insight. For example, mathematicians spent almost twenty years trying to prove or disprove the conjecture that "Every planar graph is 5-choosable"[4]. Then, in 1994, Carsten Thomassen gave an induction proof simple enough to explain on a napkin. The key turned out to be finding an extremely clever induction hypothesis; with that in hand, completing the argument is easy!

A Faulty Induction Proof

False Theorem. *All horses are the same color.*

Notice that no n is mentioned in this assertion, so we're going to have to reformulate it in a way that makes an n explicit. In particular, we'll (falsely) prove that

False Theorem 3.2.3. *In every set of $n \geq 1$ horses, all the horses are the same color.*

This a statement about all integers $n \geq 1$ rather ≥ 0, so it's natural to use a slight variation on induction: prove $P(1)$ in the base case and then prove that $P(n)$ implies $P(n+1)$ for all $n \geq 1$ in the inductive step. This is a perfectly valid variant of induction and is *not* the problem with the proof below.

False proof. The proof is by induction on n. The induction hypothesis, $P(n)$, will be

$$\text{In every set of } n \text{ horses, all are the same color.} \tag{3.6}$$

Base case: ($n = 1$). $P(1)$ is true, because in a set of horses of size 1, there's only one horse, and this horse is definitely the same color as itself.

Inductive step: Assume that $P(n)$ is true for some $n \geq 1$. that is, assume that in every set of n horses, all are the same color. Now consider a set of $n+1$ horses:

$$h_1, h_2, \ldots, h_n, h_{n+1}$$

By our assumption, the first n horses are the same color:

$$\underbrace{h_1, h_2, \ldots, h_n,}_{\text{same color}} h_{n+1}$$

Also by our assumption, the last n horses are the same color:

$$h_1, \underbrace{h_2, \ldots, h_n, h_{n+1}}_{\text{same color}}$$

[4] 5-choosability is a slight generalization of 5-colorability. Although every planar graph is 4-colorable and therefore 5-colorable, not every planar graph is 4-choosable. If this all sounds like nonsense, don't panic. We'll discuss graphs, planarity, and coloring in a later chapter.

So h_1 is the same color as the remaining horses besides h_{n+1}, and likewise h_{n+1} is the same color as the remaining horses besides h_1. So h_1 and h_{n+1} are the same color. That is, horses $h_1, h_2, \dots, h_{n+1}$ must all be the same color, and so $P(n+1)$ is true. Thus, $P(n)$ implies $P(n+1)$.

By the principle of induction, $P(n)$ is true for all $n \geq 1$. ■

We've proved something false! Is math broken? Should we all become poets? No, this proof has a mistake.

The error in this argument is in the sentence that begins, "So h_1 and h_{n+1} are the same color." The "..." notation creates the impression that there are some remaining horses besides h_1 and h_{n+1}. However, this is not true when $n = 1$. In that case, the first set is just h_1 and the second is h_2, and there are no remaining horses besides them. So h_1 and h_2 need not be the same color!

This mistake knocks a critical link out of our induction argument. We proved $P(1)$ and we *correctly* proved $P(2) \longrightarrow P(3)$, $P(3) \longrightarrow P(4)$, etc. But we failed to prove $P(1) \longrightarrow P(2)$, and so everything falls apart: we can not conclude that $P(2)$, $P(3)$, etc., are true. And, of course, these propositions are all false; there are horses of a different color.

Students sometimes claim that the mistake in the proof is because $P(n)$ is false for $n \geq 2$, and the proof assumes something false, namely, $P(n)$, in order to prove $P(n+1)$. You should think about how to explain to such a student why this claim would get no credit on a Math for Computer Science exam.

Induction versus Well Ordering

The Induction Axiom looks nothing like the Well Ordering Principle, but these two proof methods are closely related. In fact, as the examples above suggest, we can take any Well Ordering proof and reformat it into an Induction proof. Conversely, it's equally easy to take any Induction proof and reformat it into a Well Ordering proof.

So what's the difference? Well, sometimes induction proofs are clearer because they resemble recursive procedures that reduce handling an input of size $n+1$ to handling one of size n. On the other hand, Well Ordering proofs sometimes seem more natural, and also come out slightly shorter. The choice of method is really a matter of style—but style does matter.

3.2.2 Problems

Class Problems

Problem 3.6.
Use induction to prove that

$$1^3 + 2^3 + \cdots + n^3 = \left(\frac{n(n+1)}{2}\right)^2. \tag{3.7}$$

for all $n \geq 1$.

Remember to formally

1. Declare proof by induction.
2. Identify the induction hypothesis $P(n)$.
3. Establish the base case.
4. Prove that $P(n) \Rightarrow P(n+1)$.
5. Conclude that $P(n)$ holds for all $n \geq 1$.

as in the five part template.

Problem 3.7.
Prove by induction on n that

$$1 + r + r^2 + \cdots + r^n = \frac{r^{n+1} - 1}{r - 1} \tag{3.8}$$

for all $n \in \mathbb{N}$ and numbers $r \neq 1$.

Problem 3.8.
Prove by induction:

$$1 + \frac{1}{4} + \frac{1}{9} + \cdots + \frac{1}{n^2} < 2 - \frac{1}{n}, \tag{3.9}$$

for all $n > 1$.

Problem 3.9. (a) Prove by induction that a $2^n \times 2^n$ courtyard with a 1×1 statue of Bill in *a corner* can be covered with L-shaped tiles. (Do not assume or reprove the (stronger) result of Theorem 3.2.2 that Bill can be placed anywhere. The point of this problem is to show a different induction hypothesis that works.)

(b) Use the result of part (a) to prove the original claim that there is a tiling with Bill in the middle.

Problem 3.10.
Find the flaw in the following bogus proof that $a^n = 1$ for all nonnegative integers n, whenever a is a nonzero real number.

Bogus proof. The proof is by induction on n, with hypothesis

$$P(n) ::= \forall k \leq n.\, a^k = 1,$$

where k is a nonnegative integer valued variable.

Base Case: $P(0)$ is equivalent to $a^0 = 1$, which is true by definition of a^0. (By convention, this holds even if $a = 0$.)

Inductive Step: By induction hypothesis, $a^k = 1$ for all $k \in \mathbb{N}$ such that $k \leq n$. But then

$$a^{n+1} = \frac{a^n \cdot a^n}{a^{n-1}} = \frac{1 \cdot 1}{1} = 1,$$

which implies that $P(n+1)$ holds. It follows by induction that $P(n)$ holds for all $n \in \mathbb{N}$, and in particular, $a^n = 1$ holds for all $n \in \mathbb{N}$.

■

Problem 3.11.
We've proved in two different ways that

$$1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}$$

But now we're going to prove a *contradictory* theorem!

False Theorem. *For all $n \geq 0$,*

$$2 + 3 + 4 + \cdots + n = \frac{n(n+1)}{2}$$

Proof. We use induction. Let $P(n)$ be the proposition that $2 + 3 + 4 + \cdots + n = n(n+1)/2$.
Base case: $P(0)$ is true, since both sides of the equation are equal to zero. (Recall that a sum with no terms is zero.)
Inductive step: Now we must show that $P(n)$ implies $P(n+1)$ for all $n \geq 0$. So suppose that $P(n)$ is true; that is, $2 + 3 + 4 + \cdots + n = n(n+1)/2$. Then we can reason as follows:

$$\begin{aligned}
2 + 3 + 4 + \cdots + n + (n+1) &= [2 + 3 + 4 + \cdots + n] + (n+1) \\
&= \frac{n(n+1)}{2} + (n+1) \\
&= \frac{(n+1)(n+2)}{2}
\end{aligned}$$

Above, we group some terms, use the assumption $P(n)$, and then simplify. This shows that $P(n)$ implies $P(n+1)$. By the principle of induction, $P(n)$ is true for all $n \in \mathbb{N}$. ■

Where exactly is the error in this proof?

Homework Problems

Problem 3.12.

Claim 3.2.4. *If a collection of positive integers (not necessarily distinct) has sum $n \geq 1$, then the collection has product at most $3^{n/3}$.*

For example, the collection 2, 2, 3, 4, 4, 7 has the sum:

$$2+2+3+4+4+7 \quad = \quad 22$$

On the other hand, the product is:

$$\begin{aligned} 2 \cdot 2 \cdot 3 \cdot 4 \cdot 4 \cdot 7 &= 1344 \\ &\leq 3^{22/3} \\ &\approx 3154.2 \end{aligned}$$

(a) Use strong induction to prove that $n \leq 3^{n/3}$ for every integer $n \geq 0$.

(b) Prove the claim using induction or strong induction. (You may find it easier to use induction on the *number of positive integers in the collection* rather than induction on the sum n.)

Problem 3.13.

For any binary string, α, let num (α) be the nonnegative integer it represents in binary notation. For example, num $(\texttt{10}) = 2$, and num $(\texttt{0101}) = 5$.

An $n+1$-*bit adder* adds two $n+1$-bit binary numbers. More precisely, an $n+1$-bit adder takes two length $n+1$ binary strings

$$\alpha_n ::= a_n \ldots a_1 a_0,$$
$$\beta_n ::= b_n \ldots b_1 b_0,$$

and a binary digit, c_0, as inputs, and produces a length $n+1$ binary string

$$\sigma_n ::= s_n \ldots s_1 s_0,$$

and a binary digit, c_{n+1}, as outputs, and satisfies the specification:

$$\text{num}(\alpha_n) + \text{num}(\beta_n) + c_0 = 2^{n+1} c_{n+1} + \text{num}(\sigma_n). \tag{3.10}$$

There is a straighforward way to implement an $n+1$-bit adder as a digital circuit: an $n+1$-bit *ripple-carry circuit* has $1 + 2(n+1)$ binary inputs

$$a_n, \ldots, a_1, a_0, b_n, \ldots, b_1, b_0, c_0,$$

and $n+2$ binary outputs,

$$c_{n+1}, s_n, \ldots, s_1, s_0.$$

As in Problem 1.9, the ripple-carry circuit is specified by the following formulas:

$$s_i ::= a_i \text{ XOR } b_i \text{ XOR } c_i \tag{3.11}$$

$$c_{i+1} ::= (a_i \text{ AND } b_i) \text{ OR } (a_i \text{ AND } c_i) \text{ OR } (b_i \text{ AND } c_i), . \tag{3.12}$$

for $0 \leq i \leq n$.

(a) Verify that definitions (3.19) and (3.20) imply that

$$a_n + b_n + c_n = 2c_{n+1} + s_n. \tag{3.13}$$

for all $n \in \mathbb{N}$.

(b) Prove by induction on n that an $n+1$-bit ripple-carry circuit really is an $n+1$-bit adder, that is, its outputs satisfy (3.18).

Hint: You may assume that, by definition of binary representation of integers,

$$\text{num}(\alpha_{n+1}) = a_{n+1}2^{n+1} + \text{num}(\alpha_n). \tag{3.14}$$

Problem 3.14.
The 6.042 mascot, Theory Hippotamus, made a startling discovery while playing with his prized collection of unit squares over the weekend. Here is what happened.

First, Theory Hippotamus put his favorite unit square down on the floor as in Figure 3.2 (a). He noted that the length of the periphery of the resulting shape was 4, an even number. Next, he put a second unit square down next to the first so that the two squares shared an edge as in Figure 3.2 (b). He noticed that the length of the periphery of the resulting shape was now 6, which is also an even number. (The periphery of each shape in the figure is indicated by a thicker line.) Theory Hippotamus continued to place squares so that each new square shared an edge with at least one previously-placed square and no squares overlapped. Eventually, he arrived at the shape in Figure 3.2 (c). He realized that the length of the periphery of this shape was 36, which is again an even number.

Our plucky porcine pal is perplexed by this peculiar pattern. Use induction on the number of squares to prove that the length of the periphery is always even, no matter how many squares Theory Hippotamus places or how he arranges them.

3.2.2 Strong Induction

A useful variant of induction is called *strong induction*. Strong Induction and Ordinary Induction are used for exactly the same thing: proving that a predicate $P(n)$ is true for all $n \in \mathbb{N}$.

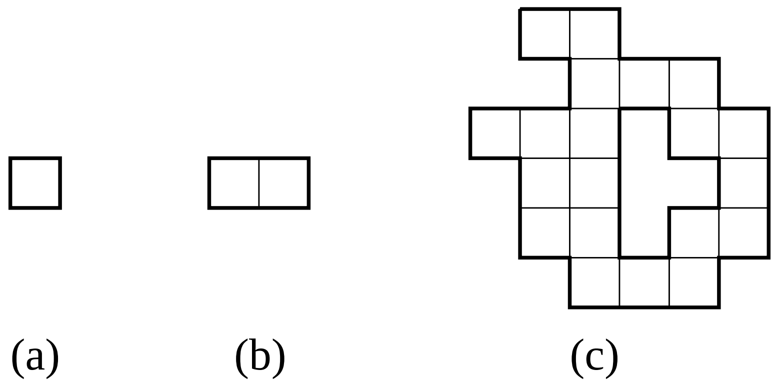


Figure 3.1: Some shapes that Theory Hippotamus created.

Principle of Strong Induction. Let $P(n)$ be a predicate. If

- $P(0)$ is true, and
- for all $n \in \mathbb{N}$, $P(0), P(1), \ldots, P(n)$ *together* imply $P(n+1)$,

then $P(n)$ is true for all $n \in \mathbb{N}$.

Rule. ***Strong Induction Rule***

$$\frac{P(0), \quad \forall n \in \mathbb{N}[(\forall m \leq n.\, P(m)) \text{ IMPLIES } P(n+1)]}{\forall n \in \mathbb{N}.\, P(n)}$$

The only change from the ordinary induction principle is that strong induction allows you to assume more stuff in the inductive step of your proof! In an ordinary induction argument, you assume that $P(n)$ is true and try to prove that $P(n+1)$ is also true. In a strong induction argument, you may assume that $P(0)$, $P(1)$, ..., and $P(n)$ are *all* true when you go to prove $P(n+1)$. These extra assumptions can only make your job easier.

Products of Primes

As a first example, we'll use strong induction to re-prove Theorem 3.1.1 which we previously proved using Well Ordering.

Lemma 3.2.5. *Every integer greater than 1 is a product of primes.*

Proof. We will prove Lemma 3.2.5 by strong induction, letting the induction hypothesis, $P(n)$, be

n is a product of primes.

So Lemma 3.2.5 will follow if we prove that $P(n)$ holds for all $n \geq 2$.

Base Case: $(n = 2)$ $P(2)$ is true because 2 is prime, and so it is a length one product of primes by convention.

Inductive step: Suppose that $n \geq 2$ and that i is a product of primes for every integer i where $2 \leq i < n + 1$. We must show that $P(n + 1)$ holds, namely, that $n + 1$ is also a product of primes. We argue by cases:

If $n + 1$ is itself prime, then it is a length one product of primes by convention, so $P(n + 1)$ holds in this case.

Otherwise, $n + 1$ is not prime, which by definition means $n + 1 = km$ for some integers k, m such that $2 \leq k, m < n + 1$. Now by strong induction hypothesis, we know that k is a product of primes. Likewise, m is a product of primes. it follows immediately that $km = n$ is also a product of primes. Therefore, $P(n + 1)$ holds in this case as well.

So $P(n + 1)$ holds in any case, which completes the proof by strong induction that $P(n)$ holds for all nonnegative integers, n.

■

Making Change

The country Inductia, whose unit of currency is the Strong, has coins worth 3Sg (3 Strongs) and 5Sg. Although the Inductians have some trouble making small change like 4Sg or 7Sg, it turns out that they can collect coins to make change for any number that is at least 8 Strongs.

Strong induction makes this easy to prove for $n+1 \geq 11$, because then $(n+1) - 3 \geq 8$, so by strong induction the Inductians can make change for exactly $(n+1) - 3$ Strongs, and then they can add a 3Sg coin to get $(n + 1)$Sg. So the only thing to do is check that they can make change for all the amounts from 8 to 10Sg, which is not too hard to do.

Here's a detailed writeup using the official format:

Proof. We prove by strong induction that the Inductians can make change for any amount of at least 8Sg. The induction hypothesis, $P(n)$ will be:

There is a collection of coins whose value is $n + 8$ Strongs.

Base case: $P(0)$ is vacuously true.

Inductive step: We assume $P(i)$ holds for all $i \leq n$, and prove that $P(n + 1)$ holds. We argue by cases:

Case $(n + 1 < 8)$: $P(n + 1)$ is vacuously true in this case.

Base case: $P(0)$ is true because a 3Sg coin together with 5Sgcoin makes 8Sg.

Inductive step: We assume $P(m)$ holds for all $m \leq n$, and prove that $P(n + 1)$ holds. We argue by cases:

Case $(n+1=1)$: We have to make $(n+1)+8=9$Sg. We can do this using three 3Sg coins.

Case $(n+1=2)$: We have to make $(n+1)+8=10$Sg. Use two 5Sg coins.

Case $(n+1 \geq 3)$: Then $0 \leq n-2 \leq n$, so by the strong induction hypothesis, the Inductians can make change for $n-2$ Strong. Now by adding a 3Sg coin, they can make change for $(n+1)$Sg.

So in any case, $P(n+1)$ is true, and we conclude by strong induction that for all $n = 0, 1, \ldots$, the Inductians can make change for $n+8$ Strong. That is, they can make change for any number of eight or more Strong.

■

The Stacking Game

Here is another exciting game that's surely about to sweep the nation `:-)` !

You begin with a stack of n boxes. Then you make a sequence of moves. In each move, you divide one stack of boxes into two nonempty stacks. The game ends when you have n stacks, each containing a single box. You earn points for each move; in particular, if you divide one stack of height $a+b$ into two stacks with heights a and b, then you score ab points for that move. Your overall score is the sum of the points that you earn for each move. What strategy should you use to maximize your total score?

As an example, suppose that we begin with a stack of $n = 10$ boxes. Then the game might proceed as follows:

Stack Heights										**Score**
$\underline{10}$										
5	$\underline{5}$									25 points
$\underline{5}$	3	2								6
$\underline{4}$	3	2	1							4
2	$\underline{3}$	2	1	2						4
$\underline{2}$	2	2	1	2	1					2
1	$\underline{2}$	2	1	2	1	1				1
1	1	$\underline{2}$	1	2	1	1	1			1
1	1	1	1	$\underline{2}$	1	1	1	1		1
1	1	1	1	1	1	1	1	1	1	1
							Total Score		=	45 points

On each line, the underlined stack is divided in the next step. Can you find a better strategy?

Analyzing the Game

Let's use strong induction to analyze the unstacking game. We'll prove that your score is determined entirely by the number of boxes —your strategy is irrelevant!

Theorem 3.2.6. *Every way of unstacking n blocks gives a score of $n(n-1)/2$ points.*

There are a couple technical points to notice in the proof:

- The template for a strong induction proof is exactly the same as for ordinary induction.
- As with ordinary induction, we have some freedom to adjust indices. In this case, we prove $P(1)$ in the base case and prove that $P(1), \ldots, P(n)$ imply $P(n+1)$ for all $n \geq 1$ in the inductive step.

Proof. The proof is by strong induction. Let $P(n)$ be the proposition that every way of unstacking n blocks gives a score of $n(n-1)/2$.

Base case: If $n = 1$, then there is only one block. No moves are possible, and so the total score for the game is $1(1-1)/2 = 0$. Therefore, $P(1)$ is true.

Inductive step: Now we must show that $P(1)$, …, $P(n)$ imply $P(n+1)$ for all $n \geq 1$. So assume that $P(1)$, …, $P(n)$ are all true and that we have a stack of $n+1$ blocks. The first move must split this stack into substacks with positive sizes a and b where $a + b = n + 1$ and $0 < a, b \leq n$. Now the total score for the game is the sum of points for this first move plus points obtained by unstacking the two resulting substacks:

$$\begin{aligned}
\text{total score} &= (\text{score for 1st move}) \\
&\quad + (\text{score for unstacking } a \text{ blocks}) \\
&\quad + (\text{score for unstacking } b \text{ blocks}) \\
&= ab + \frac{a(a-1)}{2} + \frac{b(b-1)}{2} && \text{by } P(a) \text{ and } P(b) \\
&= \frac{(a+b)^2 - (a+b)}{2} = \frac{(a+b)((a+b)-1)}{2} \\
&= \frac{(n+1)n}{2}
\end{aligned}$$

This shows that $P(1)$, $P(2)$, …, $P(n)$ imply $P(n+1)$.

Therefore, the claim is true by strong induction. ■

3.2.3 Strong Induction versus Induction

Is strong induction really "stronger" than ordinary induction? You can assume a lot more when proving the induction step, so it may seem that strong induction is much more powerful, but it's not. Strong induction may make it easier to prove a proposition, but any proof by strong induction can be reformatted to prove the same thing by ordinary induction (using a slightly more complicated induction hypothesis). Again, the choice of method is a matter of style.

When you're doing a proof by strong induction, you should say so: it will help your reader to know that $P(n+1)$ may not follow directly from just $P(n)$.

3.3 Induction versus Well Ordering

3.2.4 Problems

Practice Problems

Problem 3.15.
Find all possible (nonzero) amounts of postage that can be paid exactly using 3 and 5 cent stamps. Use induction to prove that your answer is correct.

Hint: Let $S(n)$ mean that exactly n cents of postage can be paid using only 3 and 5 cent stamps. Prove that the following proposition is true as part of your solution.

$$\forall n.\ (n \geq 8) \text{ IMPLIES } S(n).$$

Class Problems

Problem 3.16.
Use induction to prove that

$$1^3 + 2^3 + \cdots + n^3 = \left(\frac{n(n+1)}{2}\right)^2. \tag{3.15}$$

for all $n \geq 1$.

Remember to formally

1. Declare proof by induction.
2. Identify the induction hypothesis $P(n)$.
3. Establish the base case.
4. Prove that $P(n) \Rightarrow P(n+1)$.
5. Conclude that $P(n)$ holds for all $n \geq 1$.

as in the five part template.

Problem 3.17.
Prove by induction on n that

$$1 + r + r^2 + \cdots + r^n = \frac{r^{n+1} - 1}{r - 1} \tag{3.16}$$

for all $n \in \mathbb{N}$ and numbers $r \neq 1$.

Problem 3.18.
Prove by induction:

$$1 + \frac{1}{4} + \frac{1}{9} + \cdots + \frac{1}{n^2} < 2 - \frac{1}{n}, \tag{3.17}$$

for all $n > 1$.

Problem 3.19. **(a)** Prove by induction that a $2^n \times 2^n$ courtyard with a 1×1 statue of Bill in *a corner* can be covered with L-shaped tiles. (Do not assume or reprove the (stronger) result of Theorem 3.2.2 that Bill can be placed anywhere. The point of this problem is to show a different induction hypothesis that works.)

(b) Use the result of part (a) to prove the original claim that there is a tiling with Bill in the middle.

Problem 3.20.
Find the flaw in the following bogus proof that $a^n = 1$ for all nonnegative integers n, whenever a is a nonzero real number.

Bogus proof. The proof is by induction on n, with hypothesis

$$P(n) ::= \forall k \leq n.\, a^k = 1,$$

where k is a nonnegative integer valued variable.

Base Case: $P(0)$ is equivalent to $a^0 = 1$, which is true by definition of a^0. (By convention, this holds even if $a = 0$.)

Inductive Step: By induction hypothesis, $a^k = 1$ for all $k \in \mathbb{N}$ such that $k \leq n$. But then

$$a^{n+1} = \frac{a^n \cdot a^n}{a^{n-1}} = \frac{1 \cdot 1}{1} = 1,$$

which implies that $P(n+1)$ holds. It follows by induction that $P(n)$ holds for all $n \in \mathbb{N}$, and in particular, $a^n = 1$ holds for all $n \in \mathbb{N}$.

■

Problem 3.21.
We've proved in two different ways that

$$1 + 2 + 3 + \cdots + n = \frac{n(n+1)}{2}$$

But now we're going to prove a *contradictory* theorem!

False Theorem. *For all $n \geq 0$,*

$$2 + 3 + 4 + \cdots + n = \frac{n(n+1)}{2}$$

Proof. We use induction. Let $P(n)$ be the proposition that $2+3+4+\cdots+n = n(n+1)/2$.
Base case: $P(0)$ is true, since both sides of the equation are equal to zero. (Recall that a sum with no terms is zero.)
Inductive step: Now we must show that $P(n)$ implies $P(n+1)$ for all $n \geq 0$. So suppose that $P(n)$ is true; that is, $2+3+4+\cdots+n = n(n+1)/2$. Then we can reason as follows:

$$\begin{aligned} 2+3+4+\cdots+n+(n+1) &= [2+3+4+\cdots+n]+(n+1) \\ &= \frac{n(n+1)}{2}+(n+1) \\ &= \frac{(n+1)(n+2)}{2} \end{aligned}$$

Above, we group some terms, use the assumption $P(n)$, and then simplify. This shows that $P(n)$ implies $P(n+1)$. By the principle of induction, $P(n)$ is true for all $n \in \mathbb{N}$. ■

Where exactly is the error in this proof?

Problem 3.22.
Define the *potential*, $p(S)$, of a stack of blocks, S, to be $k(k-1)/2$ where k is the number of blocks in S. Define the potential, $p(A)$, of a set of stacks, A, to be the sum of the potentials of the stacks in A.

Generalize Theorem 3.2.6 about scores in the stacking game to show that for any set of stacks, A, if a sequence of moves starting with A leads to another set of stacks, B, then $p(A) \geq p(B)$, and the score for this sequence of moves is $p(A)-p(B)$.

Hint: Try induction on the number of moves to get from A to B.

Homework Problems

Problem 3.23.

Claim 3.2.1. *If a collection of positive integers (not necessarily distinct) has sum $n \geq 1$, then the collection has product at most $3^{n/3}$.*

For example, the collection 2, 2, 3, 4, 4, 7 has the sum:

$$2+2+3+4+4+7 \quad = \quad 22$$

On the other hand, the product is:

$$\begin{aligned} 2 \cdot 2 \cdot 3 \cdot 4 \cdot 4 \cdot 7 &= 1344 \\ &\leq 3^{22/3} \\ &\approx 3154.2 \end{aligned}$$

(a) Use strong induction to prove that $n \leq 3^{n/3}$ for every integer $n \geq 0$.

(b) Prove the claim using induction or strong induction. (You may find it easier to use induction on the *number of positive integers in the collection* rather than induction on the sum n.)

Problem 3.24.
For any binary string, α, let num (α) be the nonnegative integer it represents in binary notation. For example, num $(\texttt{10}) = 2$, and num $(\texttt{0101}) = 5$.

An $n+1$-*bit adder* adds two $n+1$-bit binary numbers. More precisely, an $n+1$-bit adder takes two length $n+1$ binary strings

$$\alpha_n ::= a_n \ldots a_1 a_0,$$
$$\beta_n ::= b_n \ldots b_1 b_0,$$

and a binary digit, c_0, as inputs, and produces a length $n+1$ binary string

$$\sigma_n ::= s_n \ldots s_1 s_0,$$

and a binary digit, c_{n+1}, as outputs, and satisfies the specification:

$$\text{num}\,(\alpha_n) + \text{num}\,(\beta_n) + c_0 = 2^{n+1} c_{n+1} + \text{num}\,(\sigma_n)\,. \tag{3.18}$$

There is a straighforward way to implement an $n+1$-bit adder as a digital circuit: an $n+1$-bit *ripple-carry circuit* has $1 + 2(n+1)$ binary inputs

$$a_n, \ldots, a_1, a_0, b_n, \ldots, b_1, b_0, c_0,$$

and $n+2$ binary outputs,

$$c_{n+1}, s_n, \ldots, s_1, s_0.$$

As in Problem 1.9, the ripple-carry circuit is specified by the following formulas:

$$s_i ::= a_i \;\text{ XOR }\; b_i \;\text{ XOR }\; c_i \tag{3.19}$$
$$c_{i+1} ::= (a_i \;\text{ AND }\; b_i) \text{ OR } (a_i \;\text{ AND }\; c_i) \text{ OR } (b_i \;\text{ AND }\; c_i),. \tag{3.20}$$

for $0 \leq i \leq n$.

(a) Verify that definitions (3.19) and (3.20) imply that

$$a_n + b_n + c_n = 2c_{n+1} + s_n. \tag{3.21}$$

for all $n \in \mathbb{N}$.

(b) Prove by induction on n that an $n+1$-bit ripple-carry circuit really is an $n+1$-bit adder, that is, its outputs satisfy (3.18).

Hint: You may assume that, by definition of binary representation of integers,

$$\text{num}\,(\alpha_{n+1}) = a_{n+1}2^{n+1} + \text{num}\,(\alpha_n)\,. \tag{3.22}$$

Problem 3.25.

The 6.042 mascot, Theory Hippotamus, made a startling discovery while playing with his prized collection of unit squares over the weekend. Here is what happened.

First, Theory Hippotamus put his favorite unit square down on the floor as in Figure 3.2 (a). He noted that the length of the periphery of the resulting shape was 4, an even number. Next, he put a second unit square down next to the first so that the two squares shared an edge as in Figure 3.2 (b). He noticed that the length of the periphery of the resulting shape was now 6, which is also an even number. (The periphery of each shape in the figure is indicated by a thicker line.) Theory Hippotamus continued to place squares so that each new square shared an edge with at least one previously-placed square and no squares overlapped. Eventually, he arrived at the shape in Figure 3.2 (c). He realized that the length of the periphery of this shape was 36, which is again an even number.

Our plucky porcine pal is perplexed by this peculiar pattern. Use induction on the number of squares to prove that the length of the periphery is always even, no matter how many squares Theory Hippotamus places or how he arranges them.

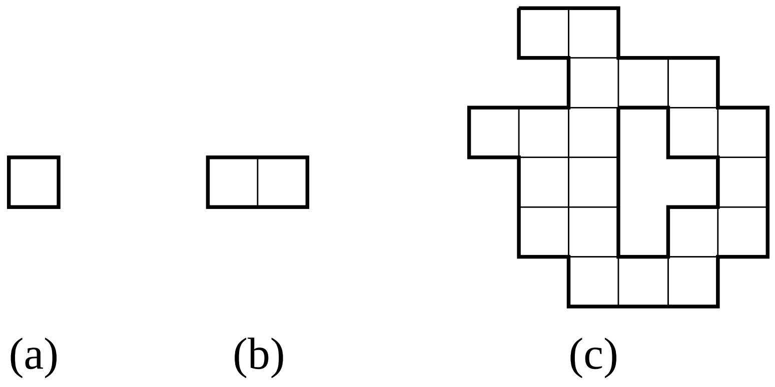


Figure 3.2: Some shapes that Theory Hippotamus created.

Problem 3.26.

Find all possible (nonzero) amounts of postage that can be paid exactly using 3 and 7 cent stamps. Use induction to prove that your answer is correct.

Hint: Let $S(n)$ mean that exactly n cents of postage can be paid using only 3 and 7 cent stamps. Prove that the following proposition is true as part of your solution.

$$\forall n.\ (n \geq 12) \text{ IMPLIES } S(n).$$

Problem 3.27.
A group of $n \geq 1$ people can be divided into teams, each containing either 4 or 7 people. What are all the possible values of n? Use induction to prove that your answer is correct.

Problem 3.28.
The following Lemma is true, but the *proof* given for it below is defective. Pinpoint *exactly* where the proof first makes an unjustified step and explain why it is unjustified.

Lemma 3.2.2. *For any prime p and positive integers $n, x_1, x_2, \ldots, x_n$, if p $\mid x_1 x_2 \ldots x_n$, then p $\mid x_i$ for some $1 \leq i \leq n$.*

False proof. Proof by strong induction on n. The induction hypothesis, $P(n)$, is that Lemma holds for n.

Base case $n = 1$: When $n = 1$, we have $p \mid x_1$, therefore we can let $i = 1$ and conclude $p \mid x_i$.

Induction step: Now assuming the claim holds for all $k \leq n$, we must prove it for $n + 1$.

So suppose $p \mid x_1 x_2 \ldots x_{n+1}$. Let $y_n = x_n x_{n+1}$, so $x_1 x_2 \ldots x_{n+1} = x_1 x_2 \ldots x_{n-1} y_n$. Since the righthand side of this equality is a product of n terms, we have by induction that p divides one of them. If $p \mid x_i$ for some $i < n$, then we have the desired i. Otherwise $p \mid y_n$. But since y_n is a product of the two terms x_n, x_{n+1}, we have by strong induction that p divides one of them. So in this case $p \mid x_i$ for $i = n$ or $i = n + 1$. ■

Chapter 4

Number Theory

Number theory is the study of the integers. *Why* anyone would want to study the integers is not immediately obvious. First of all, what's to know? There's 0, there's 1, 2, 3, and so on, and, oh yeah, -1, -2, ... Which one don't you understand? Second, what practical value is there in it? The mathematician G. H. Hardy expressed pleasure in its impracticality when he wrote:

> [Number theorists] may be justified in rejoicing that there is one science, at any rate, and that their own, whose very remoteness from ordinary human activities should keep it gentle and clean.

Hardy was specially concerned that number theory not be used in warfare; he was a pacifist. You may applaud his sentiments, but he got it wrong: Number Theory underlies modern cryptography, which is what makes secure online communication possible. Secure communication is of course crucial in war —which may leave poor Hardy spinning in his grave. It's also central to online commerce. Every time you buy a book from Amazon, check your grades on WebSIS, or use a PayPal account, you are relying on number theoretic algorithms.

4.1 Divisibility

Since we'll be focussing on properties of the integers, we'll adopt the default convention in this chapter that *variables range over integers*, $\mathbb{Z}$.

The nature of number theory emerges as soon as we consider the *divides* relation

$$a \text{ divides } b \quad \text{iff} \quad ak = b \text{ for some } k.$$

The notation, $a \mid b$, is an abbreviation for "a divides b." If $a \mid b$, then we also say that b is a *multiple* of a. A consequence of this definition is that every number divides zero.

This seems simple enough, but let's play with this definition. The Pythagoreans, an ancient sect of mathematical mystics, said that a number is *perfect* if it equals the sum of its positive integral divisors, excluding itself. For example, $6 = 1+2+3$ and $28 = 1+2+4+7+14$ are perfect numbers. On the other hand, 10 is not perfect because $1+2+5 = 8$, and 12 is not perfect because $1+2+3+4+6 = 16$. Euclid characterized all the *even* perfect numbers around 300 BC. But is there an *odd* perfect number? More than two thousand years later, we still don't know! All numbers up to about 10^{300} have been ruled out, but no one has proved that there isn't an odd perfect number waiting just over the horizon.

So a half-page into number theory, we've strayed past the outer limits of human knowledge! This is pretty typical; number theory is full of questions that are easy to pose, but incredibly difficult to answer. Interestingly, we'll see that computer scientists have found ways to turn some of these difficulties to their advantage.

Don't Panic —we're going to stick to some relatively benign parts of number theory. These super-hard unsolved problems rarely get put on exams :-) .

4.1.1 Facts About Divisibility

The lemma below states some basic facts about divisibility that are *not* difficult to prove:

Lemma 4.1.1. *The following statements about divisibility hold.*

1. *If $a \mid b$, then $a \mid bc$ for all c.*

2. *If $a \mid b$ and $b \mid c$, then $a \mid c$.*

3. *If $a \mid b$ and $a \mid c$, then $a \mid sb + tc$ for all s and t.*

4. *For all $c \neq 0$, $a \mid b$ if and only if $ca \mid cb$.*

Proof. We'll prove only part 2.; the other proofs are similar.

Proof of 2.: Since $a \mid b$, there exists an integer k_1 such that $ak_1 = b$. Since $b \mid c$, there exists an integer k_2 such that $bk_2 = c$. Substituting ak_1 for b in the second equation gives $(ak_1)k_2 = c$. So $a(k_1k_2) = c$, which implies that $a \mid c$. ■

A number $p > 1$ with no positive divisors other than 1 and itself is called a *prime*. Every other number greater than 1 is called *composite*. For example, 2, 3, 5, 7, 11, and 13 are all prime, but 4, 6, 8, and 9 are composite. Because of its special properties, the number 1 is considered to be neither prime nor composite.

Famous Conjectures in Number Theory

Fermat's Last Theorem There are no positive integers x, y, and z such that

$$x^n + y^n = z^n$$

for some integer $n > 2$. In a book he was reading around 1630, Fermat claimed to have a proof but not enough space in the margin to write it down. Wiles finally gave a proof of the theorem in 1994, after seven years of working in secrecy and isolation in his attic. His proof did not fit in any margin.

Goldbach Conjecture Every even integer greater than two equal to the sum of two primes. For example, $4 = 2 + 2$, $6 = 3 + 3$, $8 = 3 + 5$, etc. The conjecture holds for all numbers up to 10^{16}. In 1939 Schnirelman proved that every even number can be written as the sum of not more than 300,000 primes, which was a start. Today, we know that every even number is the sum of at most 6 primes.

Twin Prime Conjecture There are infinitely many primes p such that $p + 2$ is also a prime. In 1966 Chen showed that there are infinitely many primes p such that $p + 2$ is the product of at most two primes. So the conjecture is known to be *almost* true!

Primality Testing There is an efficient way to determine whether a number is prime. A naive search for factors of an integer n takes a number of steps proportional to $\sqrt{n}$, which is exponential in the *size* of n in decimal or binary notation. All known procedures for prime checking blew up like this on various inputs. Finally in 2002, an amazingly simple, new method was discovered by Agrawal, Kayal, and Saxena, which showed that prime testing only required a polynomial number of steps. Their paper began with a quote from Gauss emphasizing the importance and antiquity of the problem even in his time— two centuries ago. So prime testing is definitely not in the category of infeasible problems requiring an exponentially growing number of steps in bad cases.

Factoring Given the product of two large primes $n = pq$, there is no efficient way to recover the primes p and q. The best known algorithm is the "number field sieve", which runs in time proportional to:

$$e^{1.9(\ln n)^{1/3}(\ln \ln n)^{2/3}}$$

This is infeasible when n has 300 digits or more.

4.1.2 When Divisibility Goes Bad

As you learned in elementary school, if one number does *not* evenly divide another, you get a "quotient" and a "remainder" left over. More precisely:

Theorem 4.1.2 (Division Theorem). [1] *Let n and d be integers such that $d > 0$. Then there exists a unique pair of integers q and r, such that*

$$n = q \cdot d + r \text{ AND } 0 \leq r < d. \tag{4.1}$$

The number q is called the *quotient* and the number r is called the *remainder* of n divided by d. We use the notation $\text{qcnt}(n, d)$ for the quotient and $\text{rem}(n, d)$ for the remainder.

For example, $\text{qcnt}(2716, 10) = 271$ and $\text{rem}(2716, 10) = 6$, since $2716 = 271 \cdot 10 + 6$. Similarly, $\text{rem}(-11, 7) = 3$, since $-11 = (-2) \cdot 7 + 3$. There is a remainder operator built into many programming languages. For example, the expression "32 % 5" evaluates to 2 in Java, C, and C++. However, all these languages treat negative numbers strangely.

We'll take this familiar Division Theorem for granted without proof.

4.1.3 Die Hard

We've previously looked at the Die Hard water jug problem with jugs of sizes 3 and 5, and 3 and 9. A little number theory lets us solve all these silly water jug questions at once. In particular, it will be easy to figure out exactly which amounts of water can be measured out using jugs with capacities a and b.

Finding an Invariant Property

Suppose that we have water jugs with capacities a and b with $b \geq a$. The state of the system is described below with a pair of numbers (x, y), where x is the amount of water in the jug with capacity a and y is the amount in the jug with capacity b. Let's carry out sample operations and see what happens, assuming the b-jug is big enough:

$$\begin{aligned}
(0,0) &\to (a,0) && \text{fill first jug}\\
&\to (0,a) && \text{pour first into second}\\
&\to (a,a) && \text{fill first jug}\\
&\to (2a-b,b) && \text{pour first into second (assuming } 2a \geq b)\\
&\to (2a-b,0) && \text{empty second jug}\\
&\to (0,2a-b) && \text{pour first into second}\\
&\to (a,2a-b) && \text{fill first}\\
&\to (3a-2b,b) && \text{pour first into second (assuming } 3a \geq 2b)
\end{aligned}$$

[1] This theorem is often called the "Division Algorithm," even though it is not what we would call an algorithm.

What leaps out is that at every step, the amount of water in each jug is of the form

$$s \cdot a + t \cdot b \tag{4.2}$$

for some integers s and t. An expression of the form (4.2) is called an *integer linear combination* of a and b, but in this chapter we'll just call it a *linear combination*, since we're only talking integers. So we're suggesting:

Lemma 4.1.3. *Suppose that we have water jugs with capacities a and b. Then the amount of water in each jug is always a linear combination of a and b.*

Lemma 4.1.3 is easy to prove by induction on the number of pourings.

Proof. The induction hypothesis, $P(n)$, is the proposition that after n steps, the amount of water in each jug is a linear combination of a and b.
Base case: $(n = 0)$. $P(0)$ is true, because both jugs are initially empty, and $0 \cdot a + 0 \cdot b = 0$.
Inductive step. We assume by induction hypothesis that after n steps the amount of water in each jug is a linear combination of a and b. There are two cases:

- If we fill a jug from the fountain or empty a jug into the fountain, then that jug is empty or full. The amount in the other jug remains a linear combination of a and b. So $P(n + 1)$ holds.

- Otherwise, we pour water from one jug to another until one is empty or the other is full. By our assumption, the amount in each jug is a linear combination of a and b before we begin pouring:

$$j_1 = s_1 \cdot a + t_1 \cdot b$$
$$j_2 = s_2 \cdot a + t_2 \cdot b$$

  After pouring, one jug is either empty (contains 0 gallons) or full (contains a or b gallons). Thus, the other jug contains either $j_1 + j_2$ gallons, $j_1 + j_2 - a$, or $j_1 + j_2 - b$ gallons, all of which are linear combinations of a and b. So $P(n+1)$ holds in this case as well.

So in any case, $P(n + 1)$ follows, completing the proof by induction. ■

This theorem has an important corollary:

Corollary 4.1.4. *Bruce dies.*

Proof. In Die Hard 6, Bruce has water jugs with capacities 3 and 6 and must form 4 gallons of water. However, the amount in each jug is always of the form $3s + 6t$ by Lemma 4.1.3. This is always a multiple of 3 by Lemma 4.1.1.3, so he cannot measure out 4 gallons. ■

But Lemma 4.1.3 isn't very satisfying. We've just managed to recast a pretty understandable question about water jugs into a complicated question about linear combinations. This might not seem like progress. Fortunately, linear combinations are closely related to something more familiar, namely greatest common divisors, and these will help us solve the water jug problem.

4.2 The Greatest Common Divisor

We've already examined the Euclidean Algorithm for computing $\gcd(a, b)$, the greatest common divisor of a and b. This quantity turns out to be a very valuable piece of information about the relationship between a and b. We'll be making arguments about greatest common divisors all the time.

4.2.1 Linear Combinations and the GCD

The theorem below relates the greatest common divisor to linear combinations. This theorem is *very* useful; take the time to understand it and then remember it!

Theorem 4.2.1. *The greatest common divisor of a and b is equal to the smallest positive linear combination of a and b.*

For example, the greatest common divisor of 52 and 44 is 4. And, sure enough, 4 is a linear combination of 52 and 44:

$$6 \cdot 52 + (-7) \cdot 44 = 4$$

Furthermore, no linear combination of 52 and 44 is equal to a smaller positive integer.

Proof. By the Well Ordering Principle, there is a smallest positive linear combination of a and b; call it m. We'll prove that $m = \gcd(a, b)$ by showing both $\gcd(a, b) \leq m$ and $m \leq \gcd(a, b)$.

First, we show that $\gcd(a, b) \leq m$. Now any common divisor of a and b —that is, any c such that $c \mid a$ and $c \mid b$ —will divide both sa and tb, and therefore also divides $sa + tb$. The $\gcd(a, b)$ is by definition a common divisor of a and b, so

$$\gcd(a, b) \mid sa + tb \tag{4.3}$$

every s and t. In particular, $\gcd(a, b) \mid m$, which implies that $\gcd(a, b) \leq m$.

Now, we show that $m \leq \gcd(a, b)$. We do this by showing that $m \mid a$. A symmetric argument shows that $m \mid b$, which means that m is a common divisor of a and b. Thus, m must be less than or equal to the *greatest* common divisor of a and b.

All that remains is to show that $m \mid a$. By the Division Algorithm, there exists a quotient q and remainder r such that:

$$a = q \cdot m + r \qquad \text{(where } 0 \leq r < m\text{)}$$

Recall that $m = sa + tb$ for some integers s and t. Substituting in for m gives:

$$\begin{aligned} a &= q \cdot (sa + tb) + r, && \text{so} \\ r &= (1 - qs)a + (-qt)b. \end{aligned}$$

We've just expressed r as a linear combination of a and b. However, m is the *smallest* positive linear combination and $0 \leq r < m$. The only possibility is that the remainder r is not positive; that is, $r = 0$. This implies $m \mid a$. ■

Corollary 4.2.2. *An integer is linear combination of a and b iff it is a multiple of* $\gcd(a,b)$.

Proof. By (4.3), every linear combination of a and b is a multiple of $\gcd(a,b)$. Conversely, since $\gcd(a,b)$ is a linear combination of a and b, every multiple of $\gcd(a,b)$ is as well. ■

Now we can restate the water jugs lemma in terms of the greatest common divisor:

Corollary 4.2.3. *Suppose that we have water jugs with capacities a and b. Then the amount of water in each jug is always a multiple of* $\gcd(a,b)$.

For example, there is no way to form 2 gallons using 1247 and 899 gallon jugs, because 2 is not a multiple of $\gcd(1247, 899) = 29$.

4.2.2 Properties of the Greatest Common Divisor

We'll often make use of some basic $\gcd$ facts:

Lemma 4.2.4. *The following statements about the greatest common divisor hold:*

1. *Every common divisor of a and b divides* $\gcd(a,b)$.
2. $\gcd(ka, kb) = k \cdot \gcd(a,b)$ *for all* $k > 0$.
3. *If* $\gcd(a,b) = 1$ *and* $\gcd(a,c) = 1$, *then* $\gcd(a, bc) = 1$.
4. *If* $a \mid bc$ *and* $\gcd(a,b) = 1$, *then* $a \mid c$.
5. $\gcd(a,b) = \gcd(b, \operatorname{rem}(a,b))$.

Here's the trick to proving these statements: translate the $\gcd$ world to the linear combination world using Theorem 4.2.1, argue about linear combinations, and then translate back using Theorem 4.2.1 again.

Proof. We prove only parts 3. and 4.

Proof of 3. The assumptions together with Theorem 4.2.1 imply that there exist integers s, t, u, and v such that:

$$sa + tb = 1$$
$$ua + vc = 1$$

Multiplying these two equations gives:

$$(sa + tb)(ua + vc) = 1$$

The left side can be rewritten as $a \cdot (asu + btu + csv) + bc(tv)$. This is a linear combination of a and bc that is equal to 1, so $\gcd(a, bc) = 1$ by Theorem 4.2.1.

Proof of 4. Theorem 4.2.1 says that $\gcd(ac, bc)$ is equal to a linear combination of ac and bc. Now $a \mid ac$ trivially and $a \mid bc$ by assumption. Therefore, a divides *every* linear combination of ac and bc. In particular, a divides $\gcd(ac, bc) = c \cdot \gcd(a,b) = c \cdot 1 = c$. The first equality uses part 2. of this lemma, and the second uses the assumption that $\gcd(a,b) = 1$. ■

Lemma 4.2.4.5 is the preserved invariant from Lemma 8.3.1 that we used to prove partial correctness of the Euclidean Algorithm.

Now let's see if it's possible to make 3 gallons using 21 and 26-gallon jugs. Using Euclid's algorithm:

$$\gcd(26, 21) = \gcd(21, 5) = \gcd(5, 1) = 1.$$

Now 3 is a multiple of 1, so we can't *rule out* the possibility that 3 gallons can be formed. On the other hand, we don't know it can be done.

4.2.3 One Solution for All Water Jug Problems

Can Bruce form 3 gallons using 21 and 26-gallon jugs? This question is not so easy to answer without some number theory.

Corollary 4.2.2 says that 3 can be written as a linear combination of 21 and 26, since 3 is a multiple of $\gcd(21, 26) = 1$. In other words, there exist integers s and t such that:

$$3 = s \cdot 21 + t \cdot 26$$

We don't know what the coefficients s and t are, but we do know that they exist.

Now the coefficient s could be either positive or negative. However, we can readily transform this linear combination into an equivalent linear combination

$$3 = s' \cdot 21 + t' \cdot 26 \tag{4.4}$$

where the coefficient s' is positive. The trick is to notice that if we increase s by 26 in the original equation and decrease t by 21, then the value of the expression $s \cdot 21 + t \cdot 26$ is unchanged overall. Thus, by repeatedly increasing the value of s (by 26 at a time) and decreasing the value of t (by 21 at a time), we get a linear combination $s' \cdot 21 + t' \cdot 26 = 3$ where the coefficient s' is positive. Notice that then t' must be negative; otherwise, this expression would be much greater than 3.

Now here's how to form 3 gallons using jugs with capacities 21 and 26:

Repeat s' times:

1. Fill the 21-gallon jug.

2. Pour all the water in the 21-gallon jug into the 26-gallon jug. Whenever the 26-gallon jug becomes full, empty it out.

At the end of this process, we must have have emptied the 26-gallon jug exactly $|t'|$ times. Here's why: we've taken $s' \cdot 21$ gallons of water from the fountain, and we've poured out some multiple of 26 gallons. If we emptied fewer than $|t'|$ times, then by (4.4), the big jug would be left with at least $3 + 26$ gallons, which is more than it can hold; if we emptied it more times, the big jug would be left containing at most $3 - 26$ gallons, which is nonsense. But once we have emptied the 26-gallon jug exactly $|t'|$ times, equation (4.4) implies that there are exactly 3 gallons left.

Remarkably, we don't even need to know the coefficients s' and t' in order to use this strategy! Instead of repeating the outer loop s' times, we could just repeat *until we obtain 3 gallons*, since that must happen eventually. Of course, we have to keep track of the amounts in the two jugs so we know when we're done. Here's the solution that approach gives:

$$
\begin{array}{ccccccccc}
(0,0) & \xrightarrow{\text{fill 21}} & (21,0) & \xrightarrow{\text{pour 21 into 26}} & (0,21) & & & & \\
 & \xrightarrow{\text{fill 21}} & (21,21) & \xrightarrow{\text{pour 21 into 26}} & (16,26) & \xrightarrow{\text{empty 26}} & (16,0) & \xrightarrow{\text{pour 21 into 26}} & (0,16) \\
 & \xrightarrow{\text{fill 21}} & (21,16) & \xrightarrow{\text{pour 21 into 26}} & (11,26) & \xrightarrow{\text{empty 26}} & (11,0) & \xrightarrow{\text{pour 21 into 26}} & (0,11) \\
 & \xrightarrow{\text{fill 21}} & (21,11) & \xrightarrow{\text{pour 21 into 26}} & (6,26) & \xrightarrow{\text{empty 26}} & (6,0) & \xrightarrow{\text{pour 21 into 26}} & (0,6) \\
 & \xrightarrow{\text{fill 21}} & (21,6) & \xrightarrow{\text{pour 21 into 26}} & (1,26) & \xrightarrow{\text{empty 26}} & (1,0) & \xrightarrow{\text{pour 21 into 26}} & (0,1) \\
 & \xrightarrow{\text{fill 21}} & (21,1) & \xrightarrow{\text{pour 21 into 26}} & (0,22) & & & & \\
 & \xrightarrow{\text{fill 21}} & (21,22) & \xrightarrow{\text{pour 21 into 26}} & (17,26) & \xrightarrow{\text{empty 26}} & (17,0) & \xrightarrow{\text{pour 21 into 26}} & (0,17) \\
 & \xrightarrow{\text{fill 21}} & (21,17) & \xrightarrow{\text{pour 21 into 26}} & (12,26) & \xrightarrow{\text{empty 26}} & (12,0) & \xrightarrow{\text{pour 21 into 26}} & (0,12) \\
 & \xrightarrow{\text{fill 21}} & (21,12) & \xrightarrow{\text{pour 21 into 26}} & (7,26) & \xrightarrow{\text{empty 26}} & (7,0) & \xrightarrow{\text{pour 21 into 26}} & (0,7) \\
 & \xrightarrow{\text{fill 21}} & (21,7) & \xrightarrow{\text{pour 21 into 26}} & (2,26) & \xrightarrow{\text{empty 26}} & (2,0) & \xrightarrow{\text{pour 21 into 26}} & (0,2) \\
 & \xrightarrow{\text{fill 21}} & (21,2) & \xrightarrow{\text{pour 21 into 26}} & (0,23) & & & & \\
 & \xrightarrow{\text{fill 21}} & (21,23) & \xrightarrow{\text{pour 21 into 26}} & (18,26) & \xrightarrow{\text{empty 26}} & (18,0) & \xrightarrow{\text{pour 21 into 26}} & (0,18) \\
 & \xrightarrow{\text{fill 21}} & (21,18) & \xrightarrow{\text{pour 21 into 26}} & (13,26) & \xrightarrow{\text{empty 26}} & (13,0) & \xrightarrow{\text{pour 21 into 26}} & (0,13) \\
 & \xrightarrow{\text{fill 21}} & (21,13) & \xrightarrow{\text{pour 21 into 26}} & (8,26) & \xrightarrow{\text{empty 26}} & (8,0) & \xrightarrow{\text{pour 21 into 26}} & (0,8) \\
 & \xrightarrow{\text{fill 21}} & (21,8) & \xrightarrow{\text{pour 21 into 26}} & (3,26) & \xrightarrow{\text{empty 26}} & (3,0) & \xrightarrow{\text{pour 21 into 26}} & (0,3)
\end{array}
$$

The same approach works regardless of the jug capacities and even regardless the amount we're trying to produce! Simply repeat these two steps until the desired amount of water is obtained:

1. Fill the smaller jug.
2. Pour all the water in the smaller jug into the larger jug. Whenever the larger jug becomes full, empty it out.

By the same reasoning as before, this method eventually generates every multiple of the greatest common divisor of the jug capacities —all the quantities we can possibly produce. No ingenuity is needed at all!

4.2.4 The Pulverizer

We saw that no matter which pair of integers a and b we are given, there is always a pair of integer coefficients s and t such that

$$\gcd(a,b) = sa + tb.$$

The previous subsection gives a roundabout and inefficient method of finding such coefficients s and t. Now we describe a much quicker way to find the coefficients that dates to sixth-century India, when the method was called *kuttak*, which means "The Pulverizer."

Suppose we use Euclid's Algorithm to compute the GCD of 259 and 70, for example:

$$\begin{aligned}
\gcd(259,70) &= \gcd(70,49) && \text{since } \mathrm{rem}(259,70)=49\\
&= \gcd(49,21) && \text{since } \mathrm{rem}(70,49)=21\\
&= \gcd(21,7) && \text{since } \mathrm{rem}(49,21)=7\\
&= \gcd(7,0) && \text{since } \mathrm{rem}(21,7)=0\\
&= 7.
\end{aligned}$$

The Pulverizer goes through the same steps, but requires some extra bookkeeping along the way: as we compute $\gcd(a,b)$, we keep track of how to write each of the remainders (49, 21, and 7, in the example) as a linear combination of a and b (this is worthwhile, because our objective is to write the last nonzero remainder, which is the GCD, as such a linear combination). For our example, here is this extra bookkeeping:

x	y	$(\mathrm{rem}(x,y))$	$=$	$x - q \cdot y$
259	70	49	$=$	$259 - 3 \cdot 70$
70	49	21	$=$	$70 - 1 \cdot 49$
			$=$	$70 - 1 \cdot (259 - 3 \cdot 70)$
			$=$	$-1 \cdot 259 + 4 \cdot 70$
49	21	7	$=$	$49 - 2 \cdot 21$
			$=$	$(259 - 3 \cdot 70) - 2 \cdot (-1 \cdot 259 + 4 \cdot 70)$
			$=$	$\boxed{3 \cdot 259 - 11 \cdot 70}$
21	7	0		

We began by initializing two variables, $x = a$ and $y = b$. In the first two columns above, we carried out Euclid's algorithm. At each step, we computed $\mathrm{rem}(x,y)$, which can be written in the form $x - q \cdot y$. (Remember that the Division Algorithm says $x = q \cdot y + r$, where r is the remainder. We get $r = x - q \cdot y$ by rearranging terms.) Then we replaced x and y in this equation with equivalent linear combinations of a and b, which we already had computed. After simplifying, we were left with a linear combination of a and b that was equal to the remainder as desired. The final solution is boxed.

4.2.5 Problems

Class Problems

Problem 4.1.
A number is *perfect* if it is equal to the sum of its positive divisors, other than itself. For example, 6 is perfect, because $6 = 1 + 2 + 3$. Similarly, 28 is perfect, because $28 = 1 + 2 + 4 + 7 + 14$. Explain why $2^{k-1}(2^k - 1)$ is perfect when $2^k - 1$ is prime.[2]

[2]Euclid proved this 2300 years ago. About 250 years ago, Euler proved the converse: *every* even perfect number is of this form (for a simple proof see

Problem 4.2. **(a)** Use the Pulverizer to find integers x, y such that

$$x \cdot 50 + y \cdot 21 = \gcd(50, 21).$$

(b) Now find integers x', y' with $y' > 0$ such that

$$x' \cdot 50 + y' \cdot 21 = \gcd(50, 21)$$

Problem 4.3.
For nonzero integers, a, b, prove the following properties of divisibility and GCD'S. (You may use the fact that $\gcd(a, b)$ is an integer linear combination of a and b. You may *not* appeal to uniqueness of prime factorization because the properties below are needed to *prove* unique factorization.)

(a) Every common divisor of a and b divides $\gcd(a, b)$.

(b) If $a \mid bc$ and $\gcd(a, b) = 1$, then $a \mid c$.

(c) If $p \mid ab$ for some prime, p, then $p \mid a$ or $p \mid b$.

(d) Let m be the smallest integer linear combination of a and b that is positive. Show that $m = \gcd(a, b)$.

4.3 The Fundamental Theorem of Arithmetic

We now have almost enough tools to prove something that you probably already know.

Theorem 4.3.1 (Fundamental Theorem of Arithmetic)**.** *Every positive integer n can be written in a unique way as a product of primes:*

$$n \quad = \quad p_1 \cdot p_2 \cdots p_j \qquad\qquad (p_1 \leq p_2 \leq \cdots \leq p_j)$$

Notice that the theorem would be false if 1 were considered a prime; for example, 15 could be written as $3 \cdot 5$ or $1 \cdot 3 \cdot 5$ or $1^2 \cdot 3 \cdot 5$. Also, we're relying on a standard convention: the product of an empty set of numbers is defined to be 1, much as the sum of an empty set of numbers is defined to be 0. Without this convention, the theorem would be false for $n = 1$.

There is a certain wonder in the Fundamental Theorem, even if you've known it since you were in a crib. Primes show up erratically in the sequence of integers. In fact, their distribution seems almost random:

$$2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, \ldots$$

http://primes.utm.edu/notes/proofs/EvenPerfect.html). As is typical in number theory, apparently simple results lie at the brink of the unknown. For example, it is not known if there are an infinite number of even perfect numbers or any odd perfect numbers at all.

Basic questions about this sequence have stumped humanity for centuries. And yet we know that every natural number can be built up from primes in *exactly one way*. These quirky numbers are the building blocks for the integers. The Fundamental Theorem is not hard to prove, but we'll need a couple of preliminary facts.

Lemma 4.3.2. *If p is a prime and $p \mid ab$, then $p \mid a$ or $p \mid b$.*

Proof. The greatest common divisor of a and p must be either 1 or p, since these are the only positive divisors of p. If $\gcd(a, p) = p$, then the claim holds, because a is a multiple of p. Otherwise, $\gcd(a, p) = 1$ and so $p \mid b$ by Lemma 4.2.4.4. ■

A routine induction argument extends this statement to:

Lemma 4.3.3. *Let p be a prime. If $p \mid a_1 a_2 \cdots a_n$, then p divides some a_i.*

Now we're ready to prove the Fundamental Theorem of Arithmetic.

Proof. Theorem 3.1.1 showed, using the Well Ordering Principle, that every positive integer can be expressed as a product of primes. So we just have to prove this expression is unique. We will use Well Ordering to prove this too.

The proof is by contradiction: assume, contrary to the claim, that there exist positive integers that can be written as products of primes in more than one way. By the Well Ordering Principle, there is a smallest integer with this property. Call this integer n, and let

$$\begin{aligned} n &= p_1 \cdot p_2 \cdots p_j \\ &= q_1 \cdot q_2 \cdots q_k \end{aligned}$$

be two of the (possibly many) ways to write n as a product of primes. Then $p_1 \mid n$ and so $p_1 \mid q_1 q_2 \cdots q_k$. Lemma 4.3.3 implies that p_1 divides one of the primes q_i. But since q_i is a prime, it must be that $p_1 = q_i$. Deleting p_1 from the first product and q_i from the second, we find that n/p_1 is a positive integer *smaller* than n that can also be written as a product of primes in two distinct ways. But this contradicts the definition of n as the smallest such positive integer. ■

4.3.1 Problems

Class Problems

Problem 4.4. **(a)** Let $m = 2^9 5^{24} 11^7 17^{12}$ and $n = 2^3 7^{22} 11^{211} 13^1 17^9 19^2$. What is the $\gcd(m, n)$? What is the *least common multiple*, $\operatorname{lcm}(m, n)$, of m and n? Verify that

$$\gcd(m, n) \cdot \operatorname{lcm}(m, n) = mn. \tag{4.5}$$

(b) Describe in general how to find the $\gcd(m, n)$ and $\operatorname{lcm}(m, n)$ from the prime factorizations of m and n. Conclude that equation (4.5) holds for all positive integers m, n.

The Prime Number Theorem

Let $\pi(x)$ denote the number of primes less than or equal to x. For example, $\pi(10) = 4$ because 2, 3, 5, and 7 are the primes less than or equal to 10. Primes are very irregularly distributed, so the growth of π is similarly erratic. However, the Prime Number Theorem gives an approximate answer:

$$\lim_{x \to \infty} \frac{\pi(x)}{x / \ln x} = 1$$

Thus, primes gradually taper off. As a rule of thumb, about 1 integer out of every $\ln x$ in the vicinity of x is a prime.

The Prime Number Theorem was conjectured by Legendre in 1798 and proved a century later by de la Vallee Poussin and Hadamard in 1896. However, after his death, a notebook of Gauss was found to contain the same conjecture, which he apparently made in 1791 at age 15. (You sort of have to feel sorry for all the otherwise "great" mathematicians who had the misfortune of being contemporaries of Gauss.)

In late 2004 a billboard appeared in various locations around the country:

$$\left\{\begin{array}{c}\text{first 10-digit prime found}\\ \text{in consecutive digits of } e\end{array}\right\}\textbf{. com}$$

Substituting the correct number for the expression in curly-braces produced the URL for a Google employment page. The idea was that Google was interested in hiring the sort of people that could and would solve such a problem.

How hard is this problem? Would you have to look through thousands or millions or billions of digits of e to find a 10-digit prime? The rule of thumb derived from the Prime Number Theorem says that among 10-digit numbers, about 1 in

$$\ln 10^{10} \approx 23$$

is prime. This suggests that the problem isn't really so hard! Sure enough, the first 10-digit prime in consecutive digits of e appears quite early:

$$\begin{aligned} e =\,&2.718281828459045235360287471352662497757247093699959574966\\ &967627724076630353547594571382178525166427\mathbf{427466391}9320030\\ &599218174135966290435729003342952605956307381323286279434\ldots \end{aligned}$$

4.4 Alan Turing

The man pictured above is Alan Turing, the most important figure in the history of computer science. For decades, his fascinating life story was shrouded by government secrecy, societal taboo, and even his own deceptions.

At age 24, Turing wrote a paper entitled *On Computable Numbers, with an Application to the Entscheidungsproblem*. The crux of the paper was an elegant way to model a computer in mathematical terms. This was a breakthrough, because it allowed the tools of mathematics to be brought to bear on questions of computation. For example, with his model in hand, Turing immediately proved that there exist problems that no computer can solve— no matter how ingenious the programmer. Turing's paper is all the more remarkable because he wrote it in 1936, a full decade before any electronic computer actually existed.

The word "Entscheidungsproblem" in the title refers to one of the 28 mathematical problems posed by David Hilbert in 1900 as challenges to mathematicians of the 20th century. Turing knocked that one off in the same paper. And perhaps you've heard of the "Church-Turing thesis"? Same paper. So Turing was obviously a brilliant guy who generated lots of amazing ideas. But this lecture is about one of Turing's less-amazing ideas. It involved codes. It involved number theory. And it was sort of stupid.

Let's look back to the fall of 1937. Nazi Germany was rearming under Adolf Hitler, world-shattering war looked imminent, and— like us— Alan Turing was pondering the usefulness of number theory. He foresaw that preserving military secrets would be vital in the coming conflict and proposed a way *to encrypt communications using number theory*. This is an idea that has ricocheted up to our own time. Today, number theory is the basis for numerous public-key cryptosystems, digital signature schemes, cryptographic hash functions, and electronic payment systems. Furthermore, military funding agencies are among the biggest investors in cryptographic research. Sorry Hardy!

Soon after devising his code, Turing disappeared from public view, and half a century would pass before the world learned the full story of where he'd gone and what he did there. We'll come back to Turing's life in a little while; for now, let's investigate the code Turing left behind. The details are uncertain, since he never formally published the idea, so we'll consider a couple of possibilities.

4.4.1 Turing's Code (Version 1.0)

The first challenge is to translate a text message into an integer so we can perform mathematical operations on it. This step is not intended to make a message harder to read, so the details are not too important. Here is one approach: replace each letter of the message with two digits ($A = 01$, $B = 02$, $C = 03$, etc.) and string all the digits together to form one huge number. For example, the message "victory" could be translated this way:

$$\begin{array}{cccccccc} & \text{“v} & \text{i} & \text{c} & \text{t} & \text{o} & \text{r} & \text{y”} \\ \rightarrow & 22 & 09 & 03 & 20 & 15 & 18 & 25 \end{array}$$

Turing's code requires the message to be a prime number, so we may need to pad the result with a few more digits to make a prime. In this case, appending the digits 13 gives the number 2209032015182513, which is prime.

Now here is how the encryption process works. In the description below, m is the unencoded message (which we want to keep secret), m^* is the encrypted message (which the Nazis may intercept), and k is the key.

Beforehand The sender and receiver agree on a secret key, which is a large prime k.

Encryption The sender encrypts the message m by computing:

$$m^* = m \cdot k$$

Decryption The receiver decrypts m^* by computing:

$$\frac{m^*}{k} = \frac{m \cdot k}{k} = m$$

For example, suppose that the secret key is the prime number $k = 22801763489$ and the message m is "victory". Then the encrypted message is:

$$\begin{aligned} m^* &= m \cdot k \\ &= 2209032015182513 \cdot 22801763489 \\ &= 50369825549820718594667857 \end{aligned}$$

There are a couple of questions that one might naturally ask about Turing's code.

1. How can the sender and receiver ensure that m and k are prime numbers, as required?

   The general problem of determining whether a large number is prime or composite has been studied for centuries, and reasonably good primality tests were known even in Turing's time. In 2002, Manindra Agrawal, Neeraj Kayal, and Nitin Saxena announced a primality test that is guaranteed to work on a number n in about $(\log n)^{12}$ steps, that is, a number of steps bounded by a twelfth degree polynomial in the length (in bits) of the input, n. This definitively places primality testing way below the problems of exponential difficulty. Amazingly, the description of their breakthrough algorithm was only thirteen lines long!

   Of course, a twelfth degree polynomial grows pretty fast, so the Agrawal, *et al.* procedure is of no practical use. Still, good ideas have a way of breeding more good ideas, so there's certainly hope further improvements will lead to a procedure that is useful in practice. But the truth is, there's no practical need to improve it, since very efficient *probabilistic* procedures for prime-testing have been known since the early 1970's. These procedures have some probability of giving a wrong answer, but their probability of being wrong is so tiny that relying on their answers is the best bet you'll ever make.

2. Is Turing's code secure?

   The Nazis see only the encrypted message $m^* = m \cdot k$, so recovering the original message m requires factoring m^*. Despite immense efforts, no really efficient factoring algorithm has ever been found. It appears to be a fundamentally difficult problem, though a breakthrough someday is not impossible. In effect, Turing's code puts to practical use his discovery that there are limits to the power of computation. Thus, provided m and k are sufficiently large, the Nazis seem to be out of luck!

This all sounds promising, but there is a major flaw in Turing's code.

4.4.2 Breaking Turing's Code

Let's consider what happens when the sender transmits a *second* message using Turing's code and the same key. This gives the Nazis two encrypted messages to look at:

$$m_1^* = m_1 \cdot k \qquad \text{and} \qquad m_2^* = m_2 \cdot k$$

The greatest common divisor of the two encrypted messages, m_1^* and m_2^*, is the secret key k. And, as we've seen, the GCD of two numbers can be computed very efficiently. So after the second message is sent, the Nazis can recover the secret key and read *every* message!

It is difficult to believe a mathematician as brilliant as Turing could overlook such a glaring problem. One possible explanation is that he had a slightly different system in mind, one based on *modular* arithmetic.

4.5 Modular Arithmetic

On page 1 of his masterpiece on number theory, *Disquisitiones Arithmeticae*, Gauss introduced the notion of "congruence". Now, Gauss is another guy who managed to cough up a half-decent idea every now and then, so let's take a look at this one. Gauss said that a is *congruent* to b *modulo* n iff $n \mid (a - b)$. This is written

$$a \equiv b \pmod{n}.$$

For example:

$$29 \equiv 15 \pmod{7} \quad \text{because } 7 \mid (29 - 15).$$

There is a close connection between congruences and remainders:

Lemma 4.5.1 (Congruences and Remainders)**.**

$$a \equiv b \pmod{n} \quad \textit{iff} \quad \operatorname{rem}(a, n) = \operatorname{rem}(b, n).$$

Proof. By the Division Theorem, there exist unique pairs of integers q_1, r_1 and q_2, r_2 such that:

$$\begin{aligned} a &= q_1 n + r_1 && \text{where } 0 \leq r_1 < n, \\ b &= q_2 n + r_2 && \text{where } 0 \leq r_2 < n. \end{aligned}$$

Subtracting the second equation from the first gives:

$$a - b = (q_1 - q_2)n + (r_1 - r_2) \qquad \text{where } -n < r_1 - r_2 < n.$$

Now $a \equiv b \pmod{n}$ if and only if n divides the left side. This is true if and only if n divides the right side, which holds if and only if $r_1 - r_2$ is a multiple of n. Given the bounds on $r_1 - r_2$, this happens precisely when $r_1 = r_2$, that is, when $\operatorname{rem}(a, n) = \operatorname{rem}(b, n)$. ■

So we can also see that

$$29 \equiv 15 \pmod{7} \quad \text{because } \operatorname{rem}(29, 7) = 1 = \operatorname{rem}(15, 7).$$

This formulation explains why the congruence relation has properties like an equality relation. Notice that even though (mod 7) appears over on the right side the $\equiv$ symbol, it isn't any more strongly associated with the 15 than with the 29. It would really be clearer to write $29 \equiv_{\bmod 7} 15$ for example, but the notation with the modulus at the end is firmly entrenched and we'll stick to it.

We'll make frequent use of the following immediate Corollary of Lemma 4.5.1:

Corollary 4.5.2.

$$a \equiv \operatorname{rem}(a, n) \pmod{n}$$

Still another way to think about congruence modulo n is that it *defines a partition of the integers into n sets so that congruent numbers are all in the same set*. For example, suppose that we're working modulo 3. Then we can partition the integers into 3 sets as follows:

$$\begin{array}{lccccccccl} \{ & \ldots, & -6, & -3, & 0, & 3, & 6, & 9, & \ldots & \} \\ \{ & \ldots, & -5, & -2, & 1, & 4, & 7, & 10, & \ldots & \} \\ \{ & \ldots, & -4, & -1, & 2, & 5, & 8, & 11, & \ldots & \} \end{array}$$

according to whether their remainders on division by 3 are 0, 1, or 2. The upshot is that when arithmetic is done modulo n there are really only n different kinds of numbers to worry about, because there are only n possible remainders. In this sense, modular arithmetic is a simplification of ordinary arithmetic and thus is a good reasoning tool.

There are many useful facts about congruences, some of which are listed in the lemma below. The overall theme is that *congruences work a lot like equations*, though there are a couple of exceptions.

Lemma 4.5.3 (Facts About Congruences). *The following hold for $n \geq 1$:*

1. $a \equiv a \pmod{n}$
2. $a \equiv b \pmod{n}$ *implies* $b \equiv a \pmod{n}$
3. $a \equiv b \pmod{n}$ *and* $b \equiv c \pmod{n}$ *implies* $a \equiv c \pmod{n}$
4. $a \equiv b \pmod{n}$ *implies* $a + c \equiv b + c \pmod{n}$
5. $a \equiv b \pmod{n}$ *implies* $ac \equiv bc \pmod{n}$
6. $a \equiv b \pmod{n}$ *and* $c \equiv d \pmod{n}$ *imply* $a + c \equiv b + d \pmod{n}$
7. $a \equiv b \pmod{n}$ *and* $c \equiv d \pmod{n}$ *imply* $ac \equiv bd \pmod{n}$

Proof. Parts 1.–3. follow immediately from Lemma 4.5.1. Part 4. follows immediately from the definition that $a \equiv b \pmod{n}$ iff $n \mid (a - b)$. Likewise, part 5. follows because if $n \mid (a - b)$ then it divides $(a - b)c = ac - bc$. To prove part 6., assume

$$a \equiv b \pmod{n} \tag{4.6}$$

and

$$c \equiv d \pmod{n}. \tag{4.7}$$

Then

$$\begin{array}{lr} a + c \equiv b + c \pmod{n} & \text{(by part 4. and (4.6)),} \\ c + b \equiv d + b \pmod{n} & \text{(by part 4. and (4.7)), so} \\ b + c \equiv b + d \pmod{n} & \text{and therefore} \\ a + c \equiv b + d \pmod{n} & \text{(by part 3.)} \end{array}$$

Part 7. has a similar proof. ■

4.5.1 Turing's Code (Version 2.0)

In 1940 France had fallen before Hitler's army, and Britain alone stood against the Nazis in western Europe. British resistance depended on a steady flow of supplies brought across the north Atlantic from the United States by convoys of ships. These convoys were engaged in a cat-and-mouse game with German "U-boats" —submarines —which prowled the Atlantic, trying to sink supply ships and starve Britain into submission. The outcome of this struggle pivoted on a balance of information: could the Germans locate convoys better than the Allies could locate U-boats or vice versa?

Germany lost.

But a critical reason behind Germany's loss was made public only in 1974: Germany's naval code, *Enigma*, had been broken by the Polish Cipher Bureau (see `http://en.wikipedia.org/wiki/Polish_Cipher_Bureau`) and the secret had been turned over to the British a few weeks before the Nazi invasion of Poland in 1939. Throughout much of the war, the Allies were able to route convoys around German submarines by listening in to German communications. The British government didn't explain *how* Enigma was broken until 1996. When it was finally released (by the US), the story revealed that Alan Turing had joined the secret British codebreaking effort at Bletchley Park in 1939, where he became the lead developer of methods for rapid, bulk decryption of German Enigma messages. Turing's Enigma deciphering was an invaluable contribution to the Allied victory over Hitler.

Governments are always tight-lipped about cryptography, but the half-century of official silence about Turing's role in breaking Enigma and saving Britain may be related to some disturbing events after the war.

Let's consider an alternative interpretation of Turing's code. Perhaps we had the basic idea right (multiply the message by the key), but erred in using *conventional* arithmetic instead of *modular* arithmetic. Maybe this is what Turing meant:

Beforehand The sender and receiver agree on a large prime p, which may be made public. (This will be the modulus for all our arithmetic.) They also agree on a secret key $k \in \{1, 2, \ldots, p-1\}$.

Encryption The message m can be any integer in the set $\{0, 1, 2, \ldots, p-1\}$; in particular, the message is no longer required to be a prime. The sender encrypts the message m to produce m^* by computing:

$$m^* = \text{rem}(mk, p) \qquad (4.8)$$

Decryption (Uh-oh.)

The decryption step is a problem. We might hope to decrypt in the same way as before: by dividing the encrypted message m^* by the key k. The difficulty is that m^* is the *remainder* when mk is divided by p. So dividing m^* by k might not even give us an integer!

This decoding difficulty can be overcome with a better understanding of arithmetic modulo a prime.

4.5.2 Problems

Class Problems

Problem 4.5.
The following properties of equivalence mod n follow directly from its definition and simple properties of divisibility. See if you can prove them without looking up the proofs in the text.

(a) If $a \equiv b \pmod{n}$, then $ac \equiv bc \pmod{n}$.

(b) If $a \equiv b \pmod{n}$ and $b \equiv c \pmod{n}$, then $a \equiv c \pmod{n}$.

(c) If $a \equiv b \pmod{n}$ and $c \equiv d \pmod{n}$, then $ac \equiv bd \pmod{n}$.

(d) $\text{rem}(a, n) \equiv a \pmod{n}$.

Problem 4.6. **(a)** Why is a number written in decimal evenly divisible by 9 if and only if the sum of its digits is a multiple of 9? *Hint:* $10 \equiv 1 \pmod{9}$.

(b) Take a big number, such as 37273761261. Sum the digits, where every other one is negated:

$$3 + (-7) + 2 + (-7) + 3 + (-7) + 6 + (-1) + 2 + (-6) + 1 = -11$$

Explain why the original number is a multiple of 11 if and only if this sum is a multiple of 11.

Problem 4.7.
At one time, the Guinness Book of World Records reported that the "greatest human calculator" was a guy who could compute 13th roots of 100-digit numbers that were powers of 13. What a curious choice of tasks ....

(a) Prove that

$$d^{13} \equiv d \pmod{10} \tag{4.9}$$

for $0 \leq d < 10$.

(b) Now prove that

$$n^{13} \equiv n \pmod{10} \tag{4.10}$$

for all n.

4.6 Arithmetic with a Prime Modulus

4.6.1 Multiplicative Inverses

The *multiplicative inverse* of a number x is another number x^{-1} such that:

$$x \cdot x^{-1} = 1$$

Generally, multiplicative inverses exist over the real numbers. For example, the multiplicative inverse of 3 is $1/3$ since:

$$3 \cdot \frac{1}{3} = 1$$

The sole exception is that 0 does not have an inverse.

On the other hand, inverses generally do not exist over the integers. For example, 7 can not be multiplied by another integer to give 1.

Surprisingly, multiplicative inverses do exist when we're working *modulo a prime number*. For example, if we're working modulo 5, then 3 is a multiplicative inverse of 7, since:

$$7 \cdot 3 \equiv 1 \pmod{5}$$

(All numbers congruent to 3 modulo 5 are also multiplicative inverses of 7; for example, $7 \cdot 8 \equiv 1 \pmod{5}$ as well.) The only exception is that numbers congruent to 0 modulo 5 (that is, the multiples of 5) do not have inverses, much as 0 does not have an inverse over the real numbers. Let's prove this.

Lemma 4.6.1. *If p is prime and k is not a multiple of p, then k has a multiplicative inverse.*

Proof. Since p is prime, it has only two divisors: 1 and p. And since k is not a multiple of p, we must have $\gcd(p, k) = 1$. Therefore, there is a linear combination of p and k equal to 1:

$$sp + tk = 1$$

Rearranging terms gives:

$$sp = 1 - tk$$

This implies that $p \mid (1 - tk)$ by the definition of divisibility, and therefore $tk \equiv 1 \pmod{p}$ by the definition of congruence. Thus, t is a multiplicative inverse of k. ■

Multiplicative inverses are the key to decryption in Turing's code. Specifically, we can recover the original message by multiplying the encoded message by the *inverse* of the key:

$$\begin{aligned}
m^* \cdot k^{-1} &= \text{rem}(mk, p) \cdot k^{-1} && \text{(the def. (4.8) of } m^*\text{)} \\
&\equiv (mk)k^{-1} \pmod{p} && \text{(by Cor. 4.5.2)} \\
&\equiv m \pmod{p}.
\end{aligned}$$

This shows that m^*k^{-1} is congruent to the original message m. Since m was in the range $0, 1, \ldots, p-1$, we can recover it exactly by taking a remainder:

$$m = \text{rem}(m^*k^{-1}, p)$$

So now we can decrypt!

4.6.2 Cancellation

Another sense in which real numbers are nice is that one can cancel multiplicative terms. In other words, if we know that $m_1 k = m_2 k$, then we can cancel the k's and conclude that $m_1 = m_2$, provided $k \neq 0$. In general, cancellation is *not* valid in modular arithmetic. For example,

$$2 \cdot 3 \equiv 4 \cdot 3 \pmod{6},$$

cancelling the 3's leads to the *false* conclusion that $2 \equiv 4 \pmod{6}$. The fact that multiplicative terms can not be cancelled is the most significant sense in which congruences differ from ordinary equations. However, this difference goes away if we're working modulo a *prime*; then cancellation is valid.

Lemma 4.6.2. *Suppose p is a prime and k is not a multiple of p. Then*

$$ak \equiv bk \pmod{p} \quad \text{IMPLIES} \quad a \equiv b \pmod{p}.$$

Proof. Multiply both sides of the congruence by k^{-1}. ■

We can use this lemma to get a bit more insight into how Turing's code works. In particular, the encryption operation in Turing's code *permutes the set of possible messages*. This is stated more precisely in the following corollary.

Corollary 4.6.3. *Suppose p is a prime and k is not a multiple of p. Then the sequence:*

$$\text{rem}((1 \cdot k), p), \quad \text{rem}((2 \cdot k), p), \quad \ldots, \quad \text{rem}(((p-1) \cdot k), p)$$

is a permutation[3] *of the sequence:*

$$1, \quad 2, \quad \ldots, \quad (p-1).$$

Proof. The sequence of remainders contains $p-1$ numbers. Since $i \cdot k$ is not divisible by p for $i = 1, \ldots p-1$, all these remainders are in the range 1 to $p-1$ by the definition of remainder. Furthermore, the remainders are all different: no two numbers in the range 1 to $p-1$ are congruent modulo p, and by Lemma 4.6.2, $i \cdot k \equiv j \cdot k \pmod{p}$ if and only if $i \equiv j \pmod{p}$. Thus, the sequence of remainders must contain *all* of the numbers from 1 to $p-1$ in some order. ■

For example, suppose $p = 5$ and $k = 3$. Then the sequence:

$$\underbrace{\text{rem}((1 \cdot 3), 5)}_{=3}, \quad \underbrace{\text{rem}((2 \cdot 3), 5)}_{=1}, \quad \underbrace{\text{rem}((3 \cdot 3), 5)}_{=4}, \quad \underbrace{\text{rem}((4 \cdot 3), 5)}_{=2}$$

[3] A *permutation* of a sequence of elements is a sequence with the same elements (including repeats) possibly in a different order. More formally, if

$$\vec{e} ::= e_1, e_2, \ldots, e_n$$

is a length n sequence, and $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$ is a bijection, then

$$e_{\pi(1)}, e_{\pi(2)}, \ldots, e_{\pi(n)},$$

is a *permutation* of $\vec{e}$.

is a permutation of 1, 2, 3, 4. As long as the Nazis don't know the secret key k, they don't know how the set of possible messages are permuted by the process of encryption and thus can't read encoded messages.

4.6.3 Fermat's Little Theorem

A remaining challenge in using Turing's code is that decryption requires the inverse of the secret key k. An effective way to calculate k^{-1} follows from the proof of Lemma 4.6.1, namely

$$k^{-1} = \text{rem}(t, p)$$

where s, t are coefficients such that $sp + tk = 1$. Notice that t is easy to find using the Pulverizer.

An alternative approach, about equally efficient and probably more memorable, is to rely on Fermat's Little Theorem, which is much easier than his famous Last Theorem.

Theorem 4.6.4 (Fermat's Little Theorem)**.** *Suppose p is a prime and k is not a multiple of p. Then:*

$$k^{p-1} \equiv 1 \pmod{p}$$

Proof. We reason as follows:

$$\begin{aligned}
(p-1)! &::= 1 \cdot 2 \cdots (p-1) \\
&= \text{rem}(k, p) \cdot \text{rem}(2k, p) \cdots \text{rem}((p-1)k, p) && \text{(by Cor 4.6.3)} \\
&\equiv k \cdot 2k \cdots (p-1)k \pmod{p} && \text{(by Cor 4.5.2)} \\
&\equiv (p-1)! \cdot k^{p-1} \pmod{p} && \text{(rearranging terms)}
\end{aligned}$$

Now $(p-1)!$ is not a multiple of p because the prime factorizations of $1, 2, \ldots, (p-1)$ contain only primes smaller than p. So by Lemma 4.6.2, we can cancel $(p-1)!$ from the first and last expressions, which proves the claim. ■

Here is how we can find inverses using Fermat's Theorem. Suppose p is a prime and k is not a multiple of p. Then, by Fermat's Theorem, we know that:

$$k^{p-2} \cdot k \equiv 1 \pmod{p}$$

Therefore, k^{p-2} must be a multiplicative inverse of k. For example, suppose that we want the multiplicative inverse of 6 modulo 17. Then we need to compute $\text{rem}(6^{15}, 17)$, which we can do by successive squaring. All the congruences below hold modulo 17.

$$\begin{aligned}
6^2 &\equiv 36 \equiv 2 \\
6^4 &\equiv (6^2)^2 \equiv 2^2 \equiv 4 \\
6^8 &\equiv (6^4)^2 \equiv 4^2 \equiv 16 \\
6^{15} &\equiv 6^8 \cdot 6^4 \cdot 6^2 \cdot 6 \equiv 16 \cdot 4 \cdot 2 \cdot 6 \equiv 3
\end{aligned}$$

Therefore, $\text{rem}(6^{15}, 17) = 3$. Sure enough, 3 is the multiplicative inverse of 6 modulo 17, since:

$$3 \cdot 6 \equiv 1 \pmod{17}$$

In general, if we were working modulo a prime p, finding a multiplicative inverse by trying every value between 1 and $p-1$ would require about p operations. However, the approach above requires only about $\log p$ operations, which is far better when p is large.

4.6.4 Breaking Turing's Code —Again

The Germans didn't bother to encrypt their weather reports with the highly-secure Enigma system. After all, so what if the Allies learned that there was rain off the south coast of Iceland? But, amazingly, this practice provided the British with a critical edge in the Atlantic naval battle during 1941.

The problem was that some of those weather reports had originally been transmitted using Enigma from U-boats out in the Atlantic. Thus, the British obtained both unencrypted reports and the same reports encrypted with Enigma. By comparing the two, the British were able to determine which key the Germans were using that day and could read all other Enigma-encoded traffic. Today, this would be called a *known-plaintext attack*.

Let's see how a known-plaintext attack would work against Turing's code. Suppose that the Nazis know both m and m^* where:

$$m^* \equiv mk \pmod{p}$$

Now they can compute:

$$\begin{aligned}
m^{p-2} \cdot m^* &= m^{p-2} \cdot \text{rem}(mk, p) && \text{(def. (4.8) of } m^*\text{)}\\
&\equiv m^{p-2} \cdot mk \pmod{p} && \text{(by Cor 4.5.2)}\\
&\equiv m^{p-1} \cdot k \pmod{p} &&\\
&\equiv k \pmod{p} && \text{(Fermat's Theorem)}
\end{aligned}$$

Now the Nazis have the secret key k and can decrypt any message!

This is a huge vulnerability, so Turing's code has no practical value. Fortunately, Turing got better at cryptography after devising this code; his subsequent deciphering of Enigma messages surely saved thousands of lives, if not the whole of Britain.

4.6.5 Turing Postscript

A few years after the war, Turing's home was robbed. Detectives soon determined that a former homosexual lover of Turing's had conspired in the robbery. So they arrested him —that is, they arrested Alan Turing —because homosexuality was a British crime punishable by up to two years in prison at that time. Turing was sentenced to a hormonal "treatment" for his homosexuality: he was given estrogen injections. He began to develop breasts.

Three years later, Alan Turing, the founder of computer science, was dead. His mother explained what happened in a biography of her own son. Despite her repeated warnings, Turing carried out chemistry experiments in his own home. Apparently, her worst fear was realized: by working with potassium cyanide while eating an apple, he poisoned himself.

However, Turing remained a puzzle to the very end. His mother was a devoutly religious woman who considered suicide a sin. And, other biographers have pointed out, Turing had previously discussed committing suicide by eating a poisoned apple. Evidently, Alan Turing, who founded computer science and saved his country, took his own life in the end, and in just such a way that his mother could believe it was an accident.

Turing's last project before he disappeared from public view in 1939 involved the construction of an elaborate mechanical device to test a mathematical conjecture called the Riemann Hypothesis. This conjecture first appeared in a sketchy paper by Berhard Riemann in 1859 and is now one of the most famous unsolved problem in mathematics.

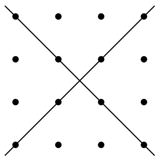
4.6.6 Problems

Class Problems

Problem 4.8.
Two nonparallel lines in the real plane intersect at a point. Algebraically, this means that the equations

$$\begin{aligned} y &= m_1x + b_1 \\ y &= m_2x + b_2 \end{aligned}$$

have a unique solution (x, y), provided $m_1 \neq m_2$. This statement would be false if we restricted x and y to the integers, since the two lines could cross at a noninteger point:



However, an analogous statement holds if we work over the integers *modulo a prime*, p. Find a solution to the congruences

$$\begin{aligned} y &\equiv m_1x + b_1 \pmod{p} \\ y &\equiv m_2x + b_2 \pmod{p} \end{aligned}$$

The Riemann Hypothesis

The formula for the sum of an infinite geometric series says:

$$1 + x + x^2 + x^3 + \cdots = \frac{1}{1-x}$$

Substituting $x = \frac{1}{2^s}$, $x = \frac{1}{3^s}$, $x = \frac{1}{5^s}$, and so on for each prime number gives a sequence of equations:

$$\begin{aligned}
1 + \frac{1}{2^s} + \frac{1}{2^{2s}} + \frac{1}{2^{3s}} + \cdots &= \frac{1}{1 - 1/2^s} \\
1 + \frac{1}{3^s} + \frac{1}{3^{2s}} + \frac{1}{3^{3s}} + \cdots &= \frac{1}{1 - 1/3^s} \\
1 + \frac{1}{5^s} + \frac{1}{5^{2s}} + \frac{1}{5^{3s}} + \cdots &= \frac{1}{1 - 1/5^s} \\
&\text{etc.}
\end{aligned}$$

Multiplying together all the left sides and all the right sides gives:

$$\sum_{n=1}^{\infty} \frac{1}{n^s} = \prod_{p \in \text{primes}} \left(\frac{1}{1 - 1/p^s} \right)$$

The sum on the left is obtained by multiplying out all the infinite series and applying the Fundamental Theorem of Arithmetic. For example, the term $1/300^s$ in the sum is obtained by multiplying $1/2^{2s}$ from the first equation by $1/3^s$ in the second and $1/5^{2s}$ in the third. Riemann noted that every prime appears in the expression on the right. So he proposed to learn about the primes by studying the equivalent, but simpler expression on the left. In particular, he regarded s as a complex number and the left side as a function, $\zeta(s)$. Riemann found that the distribution of primes is related to values of s for which $\zeta(s) = 0$, which led to his famous conjecture:

Definition 4.6.5. *The Riemann Hypothesis*: Every nontrivial zero of the zeta function $\zeta(s)$ lies on the line $s = 1/2 + ci$ in the complex plane.

A proof would immediately imply, among other things, a strong form of the Prime Number Theorem.

Researchers continue to work intensely to settle this conjecture, as they have for over a century. It is another of the Millenium Problems whose solver will earn $1,000,000 from the Clay Institute.

when $m_1 \not\equiv m_2 \pmod{p}$. Express your solution in the form $x \equiv ? \pmod{p}$ and $y \equiv ? \pmod{p}$ where the ?'s denote expressions involving m_1, m_2, b_1, and b_2. You may find it helpful to solve the original equations over the reals first.

Problem 4.9.
Let $S_k = 1^k + 2^k + \ldots + (p-1)^k$, where p is an odd prime and k is a positive multiple of $p-1$. Use Fermat's theorem to prove that $S_k \equiv -1 \pmod{p}$.

Homework Problems

Problem 4.10. (a) Use the Pulverizer to find the inverse of 13 modulo 23 in the range $\{1, \ldots, 22\}$.

(b) Use Fermat's theorem to find the inverse of 13 modulo 23 in the range $\{1, \ldots, 22\}$.

4.7 Arithmetic with an Arbitrary Modulus

Turing's code did not work as he hoped. However, his essential idea— using number theory as the basis for cryptography— succeeded spectacularly in the decades after his death.

In 1977, Ronald Rivest, Adi Shamir, and Leonard Adleman at MIT proposed a highly secure cryptosystem (called **RSA**) based on number theory. Despite decades of attack, no significant weakness has been found. Moreover, RSA has a major advantage over traditional codes: the sender and receiver of an encrypted message need not meet beforehand to agree on a secret key. Rather, the receiver has both a *secret key*, which she guards closely, and a *public key*, which she distributes as widely as possible. The sender then encrypts his message using her widely-distributed public key. Then she decrypts the received message using her closely-held private key. The use of such a *public key cryptography* system allows you and Amazon, for example, to engage in a secure transaction without meeting up beforehand in a dark alley to exchange a key.

Interestingly, RSA does not operate modulo a prime, as Turing's scheme may have, but rather modulo the product of *two* large primes. Thus, we'll need to know a bit about how arithmetic works modulo a composite number in order to understand RSA. Arithmetic modulo an arbitrary positive integer is really only a little more painful than working modulo a prime —though you may think this is like the doctor saying, "This is only going to hurt a little," before he jams a big needle in your arm.

4.7.1 Relative Primality and Phi

First, we need a new definition. Integers a and b are *relatively prime* iff $\gcd(a, b) = 1$. For example, 8 and 15 are relatively prime, since $\gcd(8, 15) = 1$. Note that, except for multiples of p, every integer is relatively prime to a prime number p.

We'll also need a certain function that is defined using relative primality. Let n be a positive integer. Then $\phi(n)$ denotes the number of integers in $\{1, 2, \ldots, n-1\}$ that are relatively prime to n. For example, $\phi(7) = 6$, since 1, 2, 3, 4, 5, and 6 are all relatively prime to 7. Similarly, $\phi(12) = 4$, since only 1, 5, 7, and 11 are relatively prime to 12. If you know the prime factorization of n, then computing $\phi(n)$ is a piece of cake, thanks to the following theorem. The function ϕ is known as *Euler's ϕ function*; it's also called Euler's *totient* function.

Theorem 4.7.1. *The function ϕ obeys the following relationships:*

(a) If a and b are relatively prime, then $\phi(ab) = \phi(a)\phi(b)$.

(b) If p is a prime, then $\phi(p^k) = p^k - p^{k-1}$ for $k \geq 1$.

Here's an example of using Theorem 4.7.1 to compute $\phi(300)$:

$$\begin{aligned}
\phi(300) &= \phi(2^2 \cdot 3 \cdot 5^2) \\
&= \phi(2^2) \cdot \phi(3) \cdot \phi(5^2) && \text{(by Theorem 4.7.1.(a))} \\
&= (2^2 - 2^1)(3^1 - 3^0)(5^2 - 5^1) && \text{(by Theorem 4.7.1.(b))} \\
&= 80.
\end{aligned}$$

The proof of Theorem 4.7.1.(a) requires a few more properties of modular arithmetic worked out in the next section (see Problem 4.15). We'll also give another a proof in a few weeks based on rules for counting things.

To prove Theorem 4.7.1.(b), notice that every pth number among the p^k numbers in the interval from 0 to $p^k - 1$ is divisible by p, and only these are divisible by p. So $1/p$th of these numbers are divisible by p and the remaining ones are not. That is,

$$\phi(p^k) = p^k - (1/p)p^k = p^k - p^{k-1}.$$

4.7.2 Generalizing to an Arbitrary Modulus

Let's generalize what we know about arithmetic modulo a prime. Now, instead of working modulo a prime p, we'll work modulo an arbitrary positive integer n. The basic theme is that arithmetic modulo n may be complicated, but the integers *relatively prime* to n remain fairly well-behaved. For example, the proof of Lemma 4.6.1 of an inverse for k modulo p extends to an inverse for k relatively prime to n:

Lemma 4.7.2. *Let n be a positive integer. If k is* relatively prime *to n, then there exists an integer k^{-1} such that:*

$$k \cdot k^{-1} \equiv 1 \pmod{n}$$

As a consequence of this lemma, we can cancel a multiplicative term from both sides of a congruence if that term is relatively prime to the modulus:

Corollary 4.7.3. *Suppose n is a positive integer and k is relatively prime to n. If*

$$ak \equiv bk \pmod{n}$$

then

$$a \equiv b \pmod{n}$$

This holds because we can multiply both sides of the first congruence by k^{-1} and simplify to obtain the second.

4.7.3 Euler's Theorem

RSA essentially relies on Euler's Theorem, a generalization of Fermat's Theorem to an arbitrary modulus. The proof is much like the proof of Fermat's Theorem, except that we focus on integers relatively prime to the modulus. Let's start with a lemma.

Lemma 4.7.4. *Suppose n is a positive integer and k is relatively prime to n. Let $k_1, \ldots, k_r$ denote all the integers relatively prime to n in the range 1 to $n-1$. Then the sequence:*

$$\text{rem}(k_1 \cdot k, n), \quad \text{rem}(k_2 \cdot k, n), \quad \text{rem}(k_3 \cdot k, n), \qquad \ldots \quad, \text{rem}(k_r \cdot k, n)$$

is a permutation of the sequence:

$$k_1, \quad k_2, \quad \ldots \quad, k_r.$$

Proof. We will show that the remainders in the first sequence are all distinct and are equal to some member of the sequence of k_j's. Since the two sequences have the same length, the first must be a permutation of the second.

First, we show that the remainders in the first sequence are all distinct. Suppose that $\text{rem}(k_i k, n) = \text{rem}(k_j k, n)$. This is equivalent to $k_i k \equiv k_j k \pmod{n}$, which implies $k_i \equiv k_j \pmod{n}$ by Corollary 4.7.3. This, in turn, means that $k_i = k_j$ since both are between 1 and $n-1$. Thus, none of the remainder terms in the first sequence is equal to any other remainder term.

Next, we show that each remainder in the first sequence equals one of the k_i. By assumption, $\gcd(k_i, n) = 1$ and $\gcd(k, n) = 1$, which means that

$$\begin{aligned} \gcd(n, \text{rem}(k_i k, n)) &= \gcd(k_i k, n) && \text{(by Lemma 4.2.4.5)} \\ &= 1 && \text{(by Lemma 4.2.4.3).} \end{aligned}$$

Now $\text{rem}(k_i k, n)$ is in the range from 0 to $n-1$ by the definition of remainder, but since it is relatively prime to n, it is actually in the range 0 to $n-1$. The k_j's are defined to be the set of all such integers, so $\text{rem}(k_i k, n)$ must equal some k_j. ■

We can now prove Euler's Theorem:

Theorem 4.7.5 (Euler's Theorem). *Suppose n is a positive integer and k is relatively prime to n. Then*

$$k^{\phi(n)} \equiv 1 \pmod{n}$$

Proof. Let $k_1, \ldots, k_r$ denote all integers relatively prime to n such that $0 \leq k_i < n$. Then $r = \phi(n)$, by the definition of the function ϕ. Now we can reason as follows:

$$\begin{aligned}
&k_1 \cdot k_2 \cdots k_r \\
&\quad = \text{rem}(k_1 \cdot k, n) \cdot \text{rem}(k_2 \cdot k, n) \cdots \text{rem}(k_r \cdot k, n) && \text{(by Lemma 4.7.4)} \\
&\quad \equiv (k_1 \cdot k) \cdot (k_2 \cdot k) \cdot \cdots (k_r \cdot k) \pmod{n} && \text{(by Cor 4.5.2)} \\
&\quad \equiv (k_1 \cdot k_2 \cdots k_r) \cdot k^r \pmod{n} && \text{(rearranging terms)}
\end{aligned}$$

Lemma 4.2.4.3. implies that $k_1 \cdot k_2 \cdots k_r$ is prime relative to n. So by Corollary 4.7.3, we can cancel this product from the first and last expressions. This proves the claim. ■

We can find multiplicative inverses using Euler's theorem as we did with Fermat's theorem: if k is relatively prime to n, then $k^{\phi(n)-1}$ is a multiplicative inverse of k modulo n. However, this approach requires computing $\phi(n)$. Unfortunately, finding $\phi(n)$ is about as hard as factoring n, and factoring is hard in general. However, when we know how to factor n, we can use Theorem 4.7.1 to compute $\phi(n)$ efficiently. Then computing $k^{\phi(n)-1}$ to find inverses is a competitive alternative to the Pulverizer.

4.7.4 RSA

Here, then, is the *RSA public key encryption scheme*:

RSA Public Key Encryption

RSA Public Key Encryption

Beforehand The receiver creates a public key and a secret key as follows.

1. Generate two distinct primes, p and q.
2. Let $n = pq$.
3. Select an integer e such that $\gcd(e, (p-1)(q-1)) = 1$.
   The *public key* is the pair (e, n). This should be distributed widely.
4. Compute d such that $de \equiv 1 \pmod{(p-1)(q-1)}$.
   The *secret key* is the pair (d, n). This should be kept hidden!

Encoding The sender encrypts message m to produce m' using the public key:

$$m' = \text{rem}(m^e, n).$$

Decoding The receiver decrypts message m' back to message m using the secret key:

$$m = \text{rem}((m')^d, n).$$

We'll explain why this way of Decoding works in Problem 4.14.

The explanation of why this way of Decoding works is worked out in Problem 4.14.

4.7.5 Problems

Practice Problems

Problem 4.11. **(a)** Prove that 22^{12001} has a multiplicative inverse modulo 175.

(b) What is the value of $\phi(175)$, where ϕ is Euler's function?

(c) What is the remainder of 22^{12001} divided by 175?

Problem 4.12. **(a)** Use the Pulverizer to find integers s, t such that

$$40s + 7t = \gcd(40, 7).$$

Show your work.

(b) Adjust your answer to part (a) to find an inverse modulo 40 of 7 in the range $\{1, \ldots, 39\}$.

Class Problems

Problem 4.13.
Let's try out RSA! There is a complete description of the algorithm at the bottom of the page. You'll probably need extra paper. **Check your work carefully!**

(a) As a team, go through the **beforehand** steps.

- Choose primes p and q to be relatively small, say in the range 10-40. In practice, p and q might contain several hundred digits, but small numbers are easier to handle with pencil and paper.
- Try $e = 3, 5, 7, \ldots$ until you find something that works. Use Euclid's algorithm to compute the gcd.
- Find d (using the Pulverizer —see appendix for a reminder on how the Pulverizer works —or Euler's Theorem).

When you're done, put your public key on the board. This lets another team send you a message.

(b) Now send an encrypted message to another team using their public key. Select your message m from the codebook below:

- 2 = Greetings and salutations!
- 3 = Yo, wassup?
- 4 = You guys are slow!
- 5 = All your base are belong to us.
- 6 = Someone on *our* team thinks someone on *your* team is kinda cute.
- 7 = You *are* the weakest link. Goodbye.

(c) Decrypt the message sent to you and verify that you received what the other team sent!

Problem 4.14.
A critical fact about RSA is, of course, that decrypting an encrypted message always gives back the original message! That is, that $\operatorname{rem}((m^d)^e, pq) = m$. This will follow from something slightly more general:

Lemma 4.7.6. *Let n be a product of distinct primes and $a \equiv 1 \pmod{\phi(n)}$ for some nonnegative integer, a. Then*

$$m^a \equiv m \pmod{n}. \tag{4.11}$$

(a) Explain why Lemma 4.7.6 implies that k and k^5 have the same last digit. For example:

$$\underline{2}^5 = 3\underline{2} \qquad 7\underline{9}^5 = 307705639\underline{9}$$

Hint: What is $\phi(10)$?

(b) Explain why Lemma 4.7.6 implies that the original message, m, equals $\text{rem}((m^e)^d, pq)$.

(c) Prove that if p is prime, then

$$m^a \equiv m \pmod{p} \tag{4.12}$$

for all nonnegative integers $a \equiv 1 \pmod{p-1}$.

(d) Prove that if n is a product of distinct primes, and $a \equiv b \pmod{p}$ for all prime factors, p, of n, then $a \equiv b \pmod{n}$.

(e) Combine the previous parts to complete the proof of Lemma 4.7.6.

Homework Problems

Problem 4.15.
Suppose m, n are relatively prime. In the problem you will prove the key property of Euler's function that $\phi(mn) = \phi(m)\phi(n)$.

(a) Prove that for any a, b, there is an x such that

$$x \equiv a \pmod{m}, \tag{4.13}$$
$$x \equiv b \pmod{n}. \tag{4.14}$$

Hint: Congruence (4.13) holds iff

$$x = jm + a. \tag{4.15}$$

for some j. So there is such an x only if

$$jm + a \equiv b \pmod{n}. \tag{4.16}$$

Solve (4.16) for j.

(b) Prove that there is an x satisfying the congruences (4.13) and (4.14) such that $0 \leq x < mn$.

(c) Prove that the x satisfying part (b) is unique.

(d) For an integer k, let k^* be the integers between 1 and $k-1$ that are relatively prime to k. Conclude from part (c) that the function

$$f : (mn)^* \to m^* \times n^*$$

defined by

$$f(x) ::= (\text{rem}(x, m), \text{rem}(x, n))$$

is a bijection.

(e) Conclude from the preceding parts of this problem that

$$\phi(mn) = \phi(m)\phi(n).$$

Exam Problems

Part II

Mathematical Data Types

Part III

Counting

Part IV

Probability