# Foreword

In my years of association with Jacob Ecco, I have had the privilege of observing him solve many important, instructive and interesting problems. I had chronicled his endeavors in *The Puzzling Adventures of Dr. Ecco.* There, I took credit for solving some of the simplest problems, and assisting him on a few others, though unwittingly most of the time.

Ecco and I both kept notebooks over that period. In his, you will find philosophical observations and witty epigrams. With his permission, I had included a few choice quotes in my chronicles. My notebook consisted mostly of scribblings as I raced in vain to keep pace with Ecco's fertile mind. However, in going over these notes, I find that they are quite helpful in understanding why his solutions work, and what might have led him to them.

It is Ecco's firm belief that the best way to learn is by solving problems. With his blessing, I am editing my notebook into a companion volume for *The Puzzling Adventures of Dr. Ecco*, so that the two together can be used as an innovative text for an introductory course in discrete mathematics.

*Justin Scarlet.*

# Preface to the second edition

*Professor Scarlet's Notebook* grew out of Andy Liu's notes for our introductory discrete mathematics course. The course was created by Andy, who exercised considerable aplomb and skill in persuading our department to offer a math course that was based on a popular puzzle book.

Currently, the course attracts close to 200 students from all disciplines, and students have requested the expanded treatment that is provided by this second edition. They have also occasionally asked for more information and background about some of the topics and this is provided in an appendix.

*The Authors.*

# Prologue

Typically, a university mathematics course presents some theory and then uses this theory to solve problems which have been chosen to illustrate it. More often than not, students complain that the problems seem contrived. We avoid this pitfall by basing this course on the problems and puzzles found in the wonderful little book *The Puzzling Adventures of Dr. Ecco.* These puzzles range from problems found in real life to problems arising in theoretical computing science, but the one thing they have in common is that they are all rooted in Discrete Mathematics.

*The Puzzling Adventures of Dr. Ecco* is a mathematical novel written by Dennis Shasha, a professor of computing science from the Courant Institute at New York University. It is not unlike *The Adventures of Sherlock Holmes*, and this notebook is a companion to Dennis Shasha's book, not unlike Dr. Watson's Journal. You are advised to read lightly through *Ecco* as early as possible to get a general feeling of what is afoot, and this should not be difficult to do, since the book is written with great humour and a fine humanistic touch.

We suggest that you resist the temptation (as great as it may be) of trying to solve all of the problems at the first reading. There will be ample time for that when we treat the problems in detail. In any case, you should always try your best to solve a problem before consulting Dr. Ecco's solution at the back of the book and before reading the corresponding discussion in this notebook.

The problems in *Ecco* vary greatly in complexity and difficulty. Sometimes you will require little more than common sense and some inqenuity to solve them; at other times, you will have to use more sophisticated approaches or even specialized methods. The ratings of the problems supplied in *Ecco's* table of contents will give you a hint about the level of difficulty of each problem.

The problems in *Ecco* can be categorized by the mathematics that can be used to help solve them. Some require coding theory, some require graph theory, some require recursion or induction, some require other branches of mathematics. Usually, it is not immediately clear to which category a particular problem belongs.

As an omniheurist for hire, Dr. Ecco has no control over the order in which his clients arrive, and problems that share a common theme are scattered haphazardly throughout the book. This notebook does not follow *Ecco's* chapters in sequence, but reorganizes them into relatively coherent modules.

Additional examples, exercises, and questions are provided in each module. The solutions to the examples are contained in the body of this notebook, those to the exercises may be found in the Appendix, while those to the questions are in a separate pamphlet available to the instructors. We do not provide an instructor's manual, since each individual instructor must tailor the course to his or her unique style. The notebook is only offered as a resource and not a paradigm.

Most of the problems we will be dealing with call upon us to do one (or both) of two things: to design an algorithm to solve the problem, or to design a mathematical structure that fits the problem. As a sneak preview, we give an example of each. **Read Sections 2.4 and 5.4** and attempt the problems there before reading on.

What is sought in Problem 1 of **Section 2.4** is a procedure or algorithm by which oil and water can be pumped alternately through the same pipe without having an inappropriate amount of either on the oil rig (too much oil accumulated or not enough water to sustain its operation). The problem specifies that oil is coming out of the rig at one barrel per minute and water is being consumed at one-tenth of a barrel per minute, with a time of six minutes to switch over from oil to water or water to oil.

It should soon become clear that for this problem the capacities of the drums are of secondary importance to the rates of transmission of the pipes. Dr. Ecco's solution should be easy to follow. Obviously, whatever procedure we come up with will be repeated periodically.

In Problem 2 of Section 2.4, the capacity of the pipe is set at 1.2 barrels per minute. This problem really asks for an optimal cycle. Let us work this through. The problem is to determine the capacities of the storage drums to support that cycle.

Suppose that a cycle lasts $x$ minutes. Since oil is coming out of the ground at 1 barrel per minute, then in that cycle we have to pump x barrels of oil to shore to keep up with production. Since we can pump 6/5 barrels per minute, we must spend $x/\frac{6}{5}$ or $5x/6$ minutes pumping oil.

On the other hand, the rig consumes $x/10$ barrels of water during the cycle. Hence we must spend $\frac{x}{10}/\frac{6}{5}$, or $x/12$ minutes pumping water. Along with the 12 minutes for change-over, we arrive at the equation $x = 5x/6 + x/12 + 12$, yielding $x = 144$.

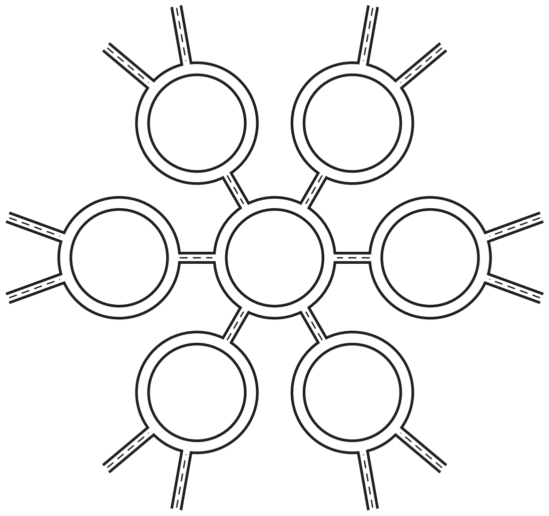
During one cycle, oil is being pumped ashore for $5x/6 = 120$ minutes and water is being sent to the rig for $x/12 = 12$ minutes, while the change-overs account for the remaining 12 minutes.

While water is being pumped in and during the change over, the amount of oil produced is $x/12 + 12 = 24$ barrels which is stored in the drum. After that, the oil is pumped to shore at the rate of 6/5 barrels per minute which moves 1 barrel of production plus 1/5 barrel of stored oil each minute. In 120 minutes this means that 120 barrels of production plus $120/5 = 24$ barrels of the stored oil is moved ashore. So actually the capacity to store 100 barrels of oil far exceeds the requirements.

While the oil is being pumped ashore and while the change-overs take place (a time totalling 132 minutes), water is being used by the rig at the rate of 1/10 barrel per minute. We need to store a enough water to sustain this, which is therefore 13.2 barrels of water. So the capacity of the water tank should be at least 13.2 barrels.

In **Section 5.4**, there is a five-way rotary. A car arrives at the rotary along one of the five two-way roads, moves counterclockwise around the rotary until it reaches the desired exit, and then leaves the rotary at that point. The *danger number* of a car entering a rotary is the number of merges it must perform both to enter and drive on that rotary, which is the number of entrances that it passes plus one for the entering merge.

We have to provide a framework for twelve two-way roads which minimizes the maximum danger number. The problem here is fairly straight-forward. An alternative solution is to have a central rotary linking six outlying ones, with two streets feeding into each of them.



At the end of some sections of *Ecco* are found **Omniheurist's Contest Puzzles**. These are stated in secret codes. Read **Section 7.2**, where you will find the last one of the ten puzzles. The featured problem there also deals with secret codes, a topic which will be treated in the third module in this notebook. If you cannot solve this problem, read the solution and see if you could have discovered it on your own.

As for the Omniheurist's Contest Puzzles, the first part of the task is to figure out what they are asking. The second and harder part is to answer them. Naturally, each Contest Puzzle is based on the corresponding sections, and you can count on some of the keywords reappearing in their statements. Solving first the tenth Contest Puzzle provides further hints. Have fun.

# Contents

# 1 Introduction

In this introductory chapter we will solve two charming puzzles that are not in *Ecco*. The solution to the first one is well-known, but even if you know it there are likely some aspects that you have not considered which may surprise you. For this particular puzzle, not much mathematics is required, but its does illustrate the general process that you should follow for all exercises. The second problem shows how a good puzzle can lead directly into some interesting and useful mathematics.

## 1.1 The general approach

**The eight-coins problem.** You are presented with eight seemingly identical coins. One of them is fake and weighs slightly more than the others. The real coins are all exactly the same weight.

You have at your disposal a pan balance, and no other weights except the coins themselves. The problem is to find the counterfeit coin in as few weighings as possible.

There is a process that you should follow with almost all of the puzzles and exercises in both *Ecco* and this notebook:

- **First, get a solution.** Some problems have more than one solution. Some solutions are more elegant than others. Don't be too concerned about getting a "neat" solution—that is something to worry about later.
- **Next, try to find out if the solution is optimal.** Sometimes a solution can be improved. If you think it cannot be improved, can you show that it can't?
- **Finally, write up your solution.** Spend some time on this. You will have notes and explanations that you have written as you worked on the problem. Organize these into a coherent presentation.

After you have written the solution, you may try to go one step further and see if you can extend the result.

The following sections show how the three guidelines come into play for the eight-coins problem.

### First, get a solution

Let us be clear—if we selected two of the coins and by good fortune one of them was the counterfeit, then weighing one against the other would reveal it. However, this really does not count as being a solution. We want something that works all the time and does not depend upon luck.

Here is a solution with three weighings that many people first come up with: Divide the eight coins into two groups of four. Weigh one group of four against the other to find out which four contain the fake. Divide that group in half, and weigh two against two to identify which pair contains the fake. For the third and final weighing compare the coins in that pair to identify the counterfeit.

This amounts to using a *binary search* which is a method of searching that repeatedly divides the data to be searched into "halves" and then determines which half contains the object that we are trying to find. At each step, the amount of data left to search is cut in half, and the procedure is quite rapid. A binary search is usually applied to linearly ordered data, but, as the eight-coins problem shows, the idea sometimes works with other data as well.

### Is the solution optimal?

Can we improve our solution to the eight-coins problem? Failing that, can we show that no improvement is possible? Is there a solution that takes fewer than three weighings?

The answer to the last question is "yes". The hint that this may be the case is to note that there are more than two possible outcomes when a balance scale is used. The left pan may contain the heavier item, the right pan may contain the heavier item, or the pans may be perfectly balanced. The counterfeit can be found with two weighings. Here's how (for want of a better term, we could call the method a *ternary* search, for it proceeds by repeatedly dividing the data into three groups):

> Split the coins into three groups of 3, 3, and 2 coins.
>
> *First weighing:* weigh 3 against 3. This tells which of the three groups has the heavy coin.
>
> *Second weighing:* If the counterfeit is in the group of two, compare one to the other. If the counterfeit is in a group of 3, leave one aside and compare the other two. This reveals which is the heavy coin.

Now we have a solution with two weighings, and again we want to either improve the solution or show that our solution is optimal.

To improve the solution, we would have to be able to locate the counterfeit with just one weighing. Can we do this? With just one weighing, either all coins are in the pans or some coins are left aside. The maximum we can leave aside is one (if we left two or more aside, the fake could be among them and we would not know which one it was). If we put more than one coin in a pan, the fake could be among them. This means that the counterfeit can be identified with one weighing only if there are two or three coins.

Your solution may undergo successive improvements before you reach a stage where it seems that nothing more can be accomplished. In an ideal world, you would always end up by showing that this "last" solution is the best possible. In real life, however, there will be times when your solution seems to be the best possible and yet you are unable to show that there is no better one. In such a case you should make a note about the situation.

### Writing up the solution

When you write up the solution, you are not obliged to explain how you arrived at the solution (and sometimes the reader does not want to know). All that you are required to do is to show that your solution works. So for the eight-coins problem, you could just explain how to do a two-weighing solution as we did above, and show that there is no one-weighing solution.

Two comments are worth making:

1. The proof that the two-weighing solution always works was clear from the description of how to carry out the procedure. Be aware, however, that there will be occasions when it will be more convenient to separate the proof from the procedure. An example of this is given in Section 1.1.1.

2. For those of you who intend to become teachers, you will have to be able to explain how you arrived at a particular solution. It would be a good idea to keep all of your work because it often contains that information.

### 1.1.1 Adaptive and nonadaptive solutions

Both solutions to the eight-coins problem — the binary search and the more efficient ternary search — are called ***adaptive*** solutions because your action at each step is adapted to the outcome from the previous one. As opposed to this, a solution in which all the actions are fully predetermined is called a ***nonadaptive*** solution.

Since an adaptive solution uses information that is discovered along the way, it is clear that an optimal adaptive solution is never worse than a nonadaptive solution. There are times, however, when a nonadaptive solution is required.

#### *A nonadaptive solution*

Here is a nonadaptive solution for the eight-coins problem. You have to keep track of the coins, so designate them A through H.

First weigh ABC against DEF, and tell me which of the three piles (ABC, DEF, or GH) is the heaviest. Then weigh ADG against BEH, and again tell me which is the heaviest pile. From this information, I can deduce which coin is the counterfeit. Here is how to do it.

Associate two labels with each coin, each label being either a zero or a one. For each weighing, label a coin with a 1 if it is in the heavy pile and with a 0 if it is in a light pile. So for each weighing, all the coins in the heavy group receive the label "1", while the rest receive the label "0". One and only one coin will be labelled with two 1's, and that is the counterfeit. The following table shows what happens when the counterfeit coin is, say, D.

| coin | 1st weighing | 2nd weighing |
|---|---|---|
| A | 0 | 1 |
| B | 0 | 0 |
| C | 0 | 0 |
| D | 1 | 1 |
| E | 1 | 0 |
| F | 1 | 0 |
| G | 0 | 1 |
| H | 0 | 0 |

Coin A is labelled 01, coin B is labelled 00, etc. The only coin that is labelled with two 1's is coin D.

Note that although a solution has been provided, we still have to show that it always works. As mentioned earlier, there is no obligation to explain what led to the solution, but we will do so in this case because it shows the solution does indeed always work.

Write the labels A through H in tabular form as shown on the right For the first weighing, compare row 1 to row 2 (ABC versus DEF). This tells us which of the three rows contains the counterfeit. For the second weighing, compare column 1 to column 2 (ADG versus BEH). This tells us which of the three columns contains the counterfeit—voila!

| | | |
|---|---|---|
| A | B | C |
| D | E | F |
| G | H | |

***Why worry about a nonadaptive solution?***

An adaptive solution is often presented as a case-by-case analysis, or as a decision tree. To explain the adaptive eight-coins solution to another person, or to program it into a computer, here's what the gory details might look like:

```
Divide the coins into groups ABC, DEF, and GH and compare ABC to DEF.
    If the coin is in group ABC, compare A to B.
        If A is the heavier, A is the counterfeit
        If B is the heavier, B is the counterfeit
        Otherwise, C is the counterfeit
    If the coin is in group DEF, compare D to E.
        If D is the heavier, D is the counterfeit
        If E is the heavier, E is the counterfeit
        Otherwise, F is the counterfeit
    Otherwise the counterfeit is in group EF, so compare E to F
        If E is the heavier, it is the counterfeit.
        Otherwise, F is the counterfeit.
```

Sometimes, if there are many steps involved, it becomes quite difficult to describe all of the cases that might arise. A nonadaptive solution is attractive if it avoids an extremely detailed case-by-case analysis.

### 1.1.2 Extending the solution

Although the problem may not require it, after you have found a solution it is only natural to try to extend it. Sometimes it is not clear what direction to try. For the eight-coins problem, there are a few questions that come to mind immediately.

1. What is the maximum number of coins such that we can find the counterfeit in 2 weighings?
2. What is the maximum number of coins such that we can find the counterfeit in $k$ weighings?
3. What is the minimum number of weighings if we have $n$ coins?
4. What happens if we don't know whether the counterfeit is heavier or lighter than the real coins, only that its weight is *different*?

The answer to the first question is pretty obvious. The same strategy (dividing into three groups) would allow us to find a single counterfeit coin among nine coins with just two weighings. This particular strategy will not work if we have ten or more coins. In fact, no strategy will locate the counterfeit with just two weighings and 10 coins. For the first weighing, the best we can do is divide the coins into three groups—one for each of the pans, and a pile left over. Dividing the 10 coins into 3 piles always produces at least one pile with 4 or more coins, so after the first weighing there is always the possibility that the counterfeit is among a group of 4 coins. Can we find a counterfeit among 4 coins with just one more weighing? No we can't: we've already answered that on page 2.

Using this idea, you can probably answer the more general questions about $k$ and $n$. Question 4 is considerably more challenging and is included in the exercise set (Question 4, page 15.)

### 1.1.3 The pigeonhole principle

In the preceding section, we mentioned that if we divide 10 coins into three piles, then at least one of the piles contains 4 coins. This is a special case of what is known as the *pigeonhole principle*. In its simplest form, the pigeonhole principle states

> If a finite number of pigeons enter a finite number of pigeonholes, and if there are more pigeons than pigeonholes, then at least one pigeonhole contains more than one pigeon.

An extended form states:

> If $p$ pigeons enter $h$ pigeonholes and if $p$ is greater than $nh$ for some integer $n$, then at least one pigeonhole contains more than $n$ pigeons.

It is easy to verify the extended version. If each of the $h$ pigeonholes contained $n$ or less pigeons then the maximum number of pigeons would be $h \cdot n$, contradicting the fact that there are *more* than $nh$ pigeons.

A discussion of the pigeonhole principle and its relationship to some other basic principles is given in the appendix. This simple principle has some very interesting consequences and here is an illustration:

**Example 1.1.1.** *Show that if eleven of the integers from 1 through 19 are selected, then some two of the selected integers sum to 20.*

In applying the pigeonhole principle, you have to describe what the pigeons are, what the pigeonholes are, and what the rule is that determines what pigeon goes into what pigeonhole. Often it is not clear what the pigeons are or what the pigeonholes are, and that is the case here.

We begin by writing down the pairs that sum to 20. They are

$$\{1,19\} \quad \{2,18\} \quad \{3,17\} \quad \{4,16\}, \quad \{5,15\} \quad \{6,14\} \quad \{7,13\} \quad \{8,12\} \quad \{9,11\}, \quad \{10,10\}$$

Then these pairs will be the pigeonholes, and the eleven chosen integers will be the pigeons. Think of the pair $\{a, b\}$ as being a box with the two labels $a$ and $b$ on it. The rule for placing the pigeons is that a pigeon goes into the box if it is one of the labels on that box. In this case there are 11 pigeons, and 10 pigeonholes, so at least two different pigeons are in the same pigeonhole. But the ones that are in the same pigeonhole sum to 20.

## 1.2 Modular Arithmetic

**The Keystone Kidnapper.** Baskerhound was at it again. This time he had kidnapped Evangeline. Ecco had sent out an urgent request to the police of Keystonia to rescue her. They had a crack swat team, but they ran into a problem. Through a sequence of blunders, they enabled Baskerhound to capture them. He has herded them and the young lady into a room containing seven caskets.

"Your deficient logic amuses me," said Baskerhound. "You have one hour to figure out how to escape from this room. Sixty minutes after I leave, six of the seven caskets will disintegrate and release a poison gas. The other casket contains the key to the door. Find the key before the hour is up and you can escape. I'll give you a clue—the key is in casket number 54321. And I've done you a favour. The caskets are not locked."

"But," said the team leader, "The caskets are numbered from 1 to 7. None of them is numbered 54321."

"I'll give you another clue: Start counting," and the kidnapper showed them how. Casket 1 was 1, Casket 2 was 2, and so on until casket 7 which was 7. Then the kidnapper started counting backwards: casket 6 was number 8, casket 5 was number 9, and so on until casket 1 which was number 13. Then the count reversed once more, and casket 2 was 14, casket 3 was 15. "You get the idea," said the kidnapper, "Goodbye," and he left them in the locked room.

"Well," said the team leader, "let's start counting."

"Hold it," said Evangeline. "We have less than 60 minutes—that's 3600 seconds, and my calculator says that 54321 divided by 3600 is about 15. We would have to count 15 caskets per second and not make a mistake."

Within a few minutes Evangeline figured out which casket contained the key. Which one was it?

Evangeline knew how to locate the critical casket because she was familiar with modular arithmetic, a subject which will be critical in understanding coding theory and cryptology.

In many instances in everyday life, we count certain entities up to a certain point and then restart the counting cycle. Examples are the hours in a day, and the days in a week. In the first case, we count in a cycle of 24, and in the second case, 7.

Suppose today is Sunday. Seven days from now it will be Sunday again. What about 1996 days from now? We can count off 7 days at a time until we arrive at a number between 0 and 6, and interpret 0 as Sunday, 1 as Monday and so on, up to 6 as Saturday. However, it is faster to just divide 1996 by 7 and look at the remainder. In a nutshell, this is what modular arithmetic is about.

We have to be a little careful with the meaning of the word "remainder". What is the remainder when 37 is divided by 7? What if $-37$ is divided by 7? Were your answers to these questions 2 and $-2$ respectively? The $-2$ is incorrect.

Here is the correct definition of remainder: Given an integer $a$ which is to be divided by a positive integer $d$, there are infinitely many integers $q$ and $r$ such that $a = q \cdot d + r$. Among all of the possible values of $r$, the ***remainder*** is the least nonnegative one. The integer $q$ that corresponds to this remainder is called the ***quotient***. For example, here are several possible values of $q$ and $r$ that satisfy $-37 = q \cdot 7 + r$:

$$\begin{array}{lll} q = -4\,, \quad r = -9 & \text{that is} & -37 = (-4) \cdot 7 + (-9) \\ q = -5\,, \quad r = -2 & & -37 = (-5) \cdot 7 + (-2) \\ q = -6\,, \quad r = +5 & & -37 = (-6) \cdot 7 + 5 \\ q = -7\,, \quad r = +12 & & -37 = (-7) \cdot 7 + 12 \end{array}$$

Although each of the above statements is true, only the the third one yields the correct quotient and remainder when $-37$ is divided by 7.

It is intuitively clear how you find the remainder when $a$ is divided by $d$: If $a$ is positive, just repeatedly subtract $d$ from $a$ and stop when you hit the first negative result. The previous result will be the remainder. For example, if we are dividing 37 by 7, the sequence is 37, 30, 23, 16, 9, 2, $-5$, so the remainder is 2.

For the case where $a$ is negative, repeatedly add $d$ to $a$ until you hit the first non-negative result. That result will be the remainder. When $-37$ is divided by 7, the sequence is $-37, -30, -23, -16, -9, -2, +5$, so the remainder is $+5$.

There is a minor problem here. If you keep halving a real number, for example, 3.0, 1.5, 0.75, 0.375, 0.1857, and so on, the process never ends. How can we be sure that the process ends when we keep subtracting $d$ from a positive integer $a$? That is, how can we be sure that among all possible nonnegative values of $r$ that satisfy

$$a = q \cdot d + r$$

there must be a smallest value?

The guarantee is provided by what is called the ***well-ordering principle***, which says precisely what we want: *Every nonempty set of nonnegative integers contains a least integer.* Although this seems to be so obvious as to hardly need stating, it is a fairly important part of number theory. One of its consequences is exactly what we need to guarantee a unique quotient and remainder. The result is called the **division algorithm** (It's not really an algorithm, but that is what its called.)

**Theorem 1.2.1 (The Division Algorithm).** *If $a$ and $d$ are integers and $d > 0$, then there are unique integers $q$ and $r$ such that $a = q \cdot d + r$, with $0 \leq r < d$.*

As in the Keystone Kidnapper puzzle, there are many cases where the remainder is more important than the quotient, and two integers $a$ and $b$ that have the same remainder when divided by the positive integer $m$ are said to be ***congruent*** to each other ***modulo*** $m$. When this happens, we write

$$a \equiv b \pmod{m},$$

If two integers are not congruent to each other modulo $m$, we write

$$a \not\equiv b \pmod{m}.$$

Congruency can be defined in other ways. The following three statements are equivalent, and any one of them may be used to define congruency modulo $m$:

1. Both integers have the same remainder when divided by $m$.
2. The difference between the two integers is divisible by $m$.
3. The two integers differ by a multiple of $m$.

Like equality, congruence is an ***equivalence relation***, meaning that it has the following important properties:

1. **Reflexivity.** For all integers $a$, $a \equiv a \pmod{m}$.
2. **Symmetry.** For all integers $a$ and $b$, if $a \equiv b \pmod{m}$ then $b \equiv a \pmod{m}$.
3. **Transitivity.** For all integers $a$, $b$, and $c$, if $a \equiv b \pmod{m}$ and $b \equiv c \pmod{m}$, then $a \equiv c \pmod{m}$.

Congruence modulo $m$ partitions the integers into $m$ ***equivalence classes***, each consisting of all integers leaving the same remainder when divided by $m$. The equivalence classes are also called ***congruence classes*** or ***residue classes***. Since the possible remainders are $0, 1, \ldots, m-1$, they are used as the standard class representatives. These are the smallest nonnegative integers from each of the classes, and they are frequently called the ***least nonnegative residues***. Sometimes we represent the entire class by enclosing one of its members in square brackets, so the congruence classes modulo 5 are [0], [1], [2], [3], and [4], that is

$$\begin{array}{rcllllllll}
[0] & = & \{\ldots, & -10, & -5, & 0, & 5, & 10, & \ldots\} \\
[1] & = & \{\ldots, & -9, & -4, & 1, & 6, & 11, & \ldots\} \\
[2] & = & \{\ldots, & -8, & -3, & 2, & 7, & 12, & \ldots\} \\
[3] & = & \{\ldots, & -7, & -2, & 3, & 8, & 13, & \ldots\} \\
[4] & = & \{\ldots, & -6, & -1, & 4, & 9, & 14, & \ldots\}
\end{array}$$

We prefer to use the standard representative from each class, but there is no hard and fast rule about this—for example, for the modulo-5 class [2] you could use [12] or [−3] instead.

Addition, subtraction and multiplication in modular arithmetic are very much like the operations in ordinary arithmetic. Congruences are handled very much like equations. Recall that you can add equations, multiply equations by a nonzero constant, and so on.

**Theorem 1.2.2.** *Suppose that* $a \equiv b \pmod{m}$ *and* $c \equiv d \pmod{m}$. *Then*

$$a \pm c \equiv b \pm d \pmod{m} \quad \textit{and} \quad ac \equiv bd \pmod{m}.$$

The preceding could be summarized by saying that if you take any $x$ from $[a]$ and any $y$ from $[c]$ then $x + y$ is always in the class $[a + c]$ and $xy$ is always in the class $[ac]$. Because of this consistency, we can conceive of adding and multiplying equivalence classes as shown in the table below.

Addition and multiplication in modulo-5 arithmetic.

| + | [0] | [1] | [2] | [3] | [4] |
|---|---|---|---|---|---|
| [0] | [0] | [1] | [2] | [3] | [4] |
| [1] | [1] | [2] | [3] | [4] | [0] |
| [2] | [2] | [3] | [4] | [0] | [1] |
| [3] | [3] | [4] | [0] | [1] | [2] |
| [4] | [4] | [0] | [1] | [2] | [3] |

| × | [0] | [1] | [2] | [3] | [4] |
|---|---|---|---|---|---|
| [0] | [0] | [0] | [0] | [0] | [0] |
| [1] | [0] | [1] | [2] | [3] | [4] |
| [2] | [0] | [2] | [4] | [1] | [3] |
| [3] | [0] | [3] | [1] | [4] | [2] |
| [4] | [0] | [4] | [3] | [2] | [1] |

More generally, when working with congruences where integers are raised to positive integer powers, like $3^n + 13 \equiv 15^k \pmod{m}$, the integers (in this case, 3, 13 and 15) can each be replaced by any other integer from the same residue class.

**Example 1.2.3.** *Solve the congruence* $(4)^6 x + 23^2 \equiv 21^7 \pmod{5}$.

*Solution.* In modulo-5 arithmetic, $4 \equiv -1$, $23 \equiv 3$, and $21 \equiv 1$, so the congruence can be rewritten as:

$$(-1)^6 x + 3^2 \equiv (1)^7 \pmod{5}.$$

Consequently $x \equiv -8 \equiv 2 \pmod{5}$. □

## The Cancellation Law

There is one significant difference between congruences and equations. In an equation like $5x = 5y$, we can cancel the 5's and conclude that $x = y$. With congruences, you have to be more careful. Consider the following congruence.

$$4 \cdot 17 \equiv 4 \cdot 8 \pmod{9}.$$

This is a valid congruence because $68 - 32 = 36$ which is divisible by 9.

If we cancel the 4's we get $17 \equiv 8 \pmod{9}$, which is again valid. Now watch what happens when we change the modulus to 6. The congruence becomes

$$4 \cdot 17 \equiv 4 \cdot 8 \pmod{6}.$$

This is a valid congruence because $68 - 32 = 36$ which is divisible by 6. However when we cancel the 4's we get $17 \equiv 8 \pmod{6}$, **which is no longer a valid congruence!** Although the algebra of congruences mimics that of equations as far as addition, subtraction, and multiplication are concerned, we have to be much more careful with division and cancellation.

**Example 1.2.4.** *In each of the following cases, give examples of integers $x$ and $y$ for which the cancellation law will fail:*

| | | | |
|---|---|---|---|
| (a) | $3x \equiv 3y \pmod{6}$ | (b) | $3x \equiv 3y \pmod{12}$ |
| (c) | $12x \equiv 12y \pmod{8}$ | (d) | $10x \equiv 10y \pmod{15}$ |

*Solution.* Using the notation $(x, y) = (a, b)$ to mean that $x = a$ and $y = b$, some possibilities are:

| | | | |
|---|---|---|---|
| (a) | $(x, y) = (4, 2)$ | (b) | $(x, y) = (8, 4)$ |
| (c) | $(x, y) = (3, 1)$ | (d) | $(x, y) = (4, 1)$ |

There are many other solutions. □

For the two congruences

$$4a \equiv 4b \pmod{9} \tag{1.1}$$

and

$$4a \equiv 4b \pmod{6} \tag{1.2}$$

it will always be true that $a \equiv b \pmod{9}$, but it may fail to be true that $a \equiv b \pmod{6}$. In other words, the cancellation law holds for congruence (1.1), but does not always hold for congruence (1.2).

Since the congruences (1.1) and (1.2) differ only in their moduli, the problem must lie with the relationship between the integer 4 and the two moduli. In (1.1), the numbers 4 and 9 have only the common factor 1, while in (1.2) the numbers 4 and 6 have a common factor of 2.

To understand what is going on here, you need a bit of Number Theory. We will state the important parts, but the proofs will be relegated to an appendix.

Given an integer $n$, any positive integer $d$ that divides $n$ without leaving a remainder is called a ***divisor*** of $n$. In this case $n$ is called a ***multiple*** of $d$. We also express the same thing by saying that $d$ ***divides*** $n$, or that $n$ is ***divisible*** by $d$, or that $d$ is a ***factor*** of $n$.

The notation used is "$d \mid n$", and it is read "$d$ divides $n$".

Given two integers $m$ and $n$, any *positive* integer that divides both $m$ and $n$ is called a ***common divisor*** of $m$ and $m$. The largest positive integer that divides both $m$ and $n$ is called the ***greatest common divisor*** of $m$ and $n$, and is denoted $\gcd(m, n)$. For example $\gcd(4, 5) = 1$ and $\gcd(4, 6) = 2$.

If the only common divisor of $m$ and $n$ is 1, then we say that $m$ and $n$ are ***relatively prime***—thus 4 and 9 are relatively prime but 4 and 6 are not.

**Theorem 1.2.5.** *If $a$ and $m$ are relatively prime integers, then the cancellation law holds for the congruence $ax \equiv ay \pmod{m}$.*
*If $a$ and $m$ are not relatively prime, then the cancellation law may fail.*

A proof is in the appendix.

## Division

In ordinary arithmetic, dividing by 5 is the same as multiplying by $\frac{1}{5}$ and dividing by $\frac{1}{5}$ is the same as multiplying by 5. In ordinary arithmetic, the numbers $\frac{1}{5}$ and 5 are called ***reciprocals***, or ***multiplicative inverses*** of each other. In general, two numbers $a$ and $b$ are said to be ***multiplicative inverses*** of each other if $ab = 1$. The same terminology is used in modular arithmetic. The two numbers $a$ and $b$ are said to be ***multiplicative inverses*** of each other modulo $m$ if $ab$ belongs to the residue class $[1]$. For example, in modulo-9 arithmetic, 4 and 7 are inverses of each other because $4 \cdot 7 = 28 \equiv 1 \pmod{9}$. In modulo-9 arithmetic multiplying by 4 amounts to "dividing" by 7, so we can define division in modular arithmetic as "multiplying by the multiplicative inverse"

**Example 1.2.6.** *Solve the congruence $7x \equiv 13 \pmod{9}$.*

*Solution.* Since 4 is the inverse of 7, multiply both sides of the congruence by 4:

$$\begin{aligned} 4(7x) &\equiv 4(4) \pmod{9}, \\ \text{so} \qquad (4 \cdot 7)x &\equiv 16 \pmod{9}, \end{aligned}$$

and since $4 \cdot 7 \equiv 1$ and $16 \equiv 7 \pmod{9}$, we get:

$$x \equiv 7 \pmod{9}. \qquad \square$$

In ordinary arithmetic, division by zero is not allowed. In modular arithmetic there may be other forbidden numbers. Below is the multiplication table for the modulo-9 residue classes:

| $\times$ | [1] | [2] | [3] | [4] | [5] | [6] | [7] | [8] |
|---|---|---|---|---|---|---|---|---|
| [1] | [1] | [2] | [3] | [4] | [5] | [6] | [7] | [8] |
| [2] | [2] | [4] | [6] | [8] | [1] | [3] | [5] | [7] |
| [3] | [3] | [6] | [0] | [3] | [6] | [0] | [3] | [6] |
| [4] | [4] | [8] | [3] | [7] | [2] | [6] | [1] | [5] |
| [5] | [5] | [1] | [6] | [2] | [7] | [3] | [8] | [4] |
| [6] | [6] | [3] | [0] | [6] | [3] | [0] | [6] | [3] |
| [7] | [7] | [5] | [3] | [1] | [8] | [6] | [4] | [2] |
| [8] | [8] | [7] | [6] | [5] | [4] | [3] | [2] | [1] |

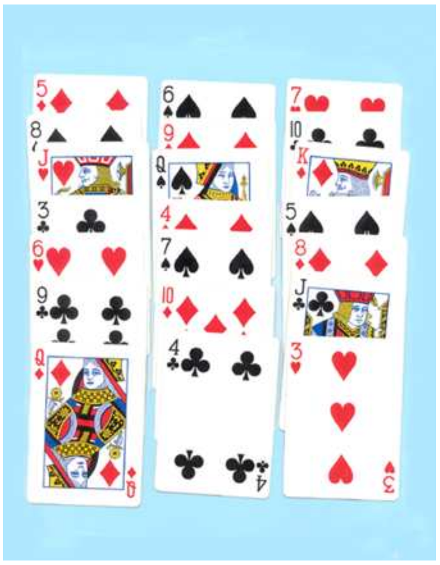
In the table we see that $[4] \times [7] = [1]$, which means that in modulo-9 arithmetic, the integers from residue class [4] are all inverses of the integers from class [7]. Similarly the integers congruent to 2 are the inverses of those that are congruent to 5. The table reveals that *in modulo-9 arithmetic, not all integers have an inverse*—the numbers congruent to 3 or 6 do not have inverses and this is because neither 3 nor 6 are relatively prime to 9.

## 1.3 Exercises

1. Here's an old trick. Take 21 cards from a standard deck. Deal them face up on the table row by row, forming three columns that are each 7 cards long. Ask a spectator to think of one of the cards and to identify the column that contains it. Scoop the cards up column by column, picking up the identified column second.

   Now deal the cards again as you did the first time. Ask the spectator to identify the column that now contains the chosen card. Scoop the cards up column by column, picking up the identified column second.

   Deal the cards once more in the same fashion, and ask the spectator to identify the column containing the chosen card. Which card in that column is the spectator's chosen card? Explain.



2. On Monday, a pharmacist received a shipment of 10 different packages of coloured vitamin pills. Each package contained pills of the same colour. On Tuesday, before selling any pills, the pharmacy received notice that one package of pills was defective—every pill in the package was contaminated. It would have been no problem if the manufacturer knew the colour of the contaminated pills, but that information was lost. However, it is known that the contaminated pills all weighed 10.0001 grams while the safe ones weighed exactly 10 grams. With one weighing on a very precise scale (not a pan balance), how can the pharmacist find out the colour of the contaminated pills?

3. Let $n$ be any positive integer and let $s$ be the sum of the digits of $n$. Show that $n$ and $s$ have the same remainder when divided by 3.

4. Let $n$ be any positive integer, and let $s$ be the alternating sum of its digits beginning with the units digit. (For example if $n = 12345$ then $s = 5 - 4 + 3 - 2 + 1 = 3$.) Show that $n$ and $s$ have the same remainder when divided by 11.

## 1.4 Questions

1. There are $n$ coins, all identical except one which is heavier than all the rest. You wish to identify the counterfeit and you may take up to three weighings using a pan balance. The only weights you are allowed are the coins themselves. Find the maximum value for $n$ and describe an adaptive procedure which will identify the counterfeit.

2. Same as the previous question, except describe a nonadaptive procedure.

3. The inspectors of fair trading found that a wholesaler of golfing equipment was swindling his retailers by including one box of substandard golf balls to every nine boxes of top grade balls he sold them. Each box contained 6 golf balls, and the external appearance of all the balls was identical. However, the substandard balls were each 1 gram too light. The retailers were informed of this discrepancy. The boxes all arrived in packs of ten, each with one substandard box - but which one?

   Phoebe Fivewood, the professional at a prestigious golf course, had just taken delivery of a large order and needed to identify the defective ones quickly. She soon found a way to do this using a pair of scales (not pan balances) which required only one weighing on each scale for each batch of ten boxes. How did she do it? Note that she did not need to know what a golf ball should weigh.

4. There are 12 coins, all identical except that one is counterfeit and is a different weight than the others. It is not known whether the counterfeit is heavier or lighter. Show how to find the counterfeit in three weighings using a pan balance. The only weights you are allowed to use are the coins themselves. [**Note.** This is a classic problem, and a nonadaptive solution was first provided by Martin Gardner. You can find the nonadaptive solution on the web. You should be able to find your own adaptive solution without searching on the web.]

5. A test for divisibility by 7, 11, or 13. Given a positive integer $n$, partition the digits of the integer into groups of three beginning at the right. (For example, if $n = 9876543210$ the partitioning is 9|876|543|210.) Let $s$ be the alternating sum of these three digit numbers, starting on the right. ($s = 210 - 543 + 876 - 9 = 534$.) Show that $n$ and $s$ have the same remainder when divided by either 7, 11, or 13. (The remainder when 9876543210 is divided by 13 is 1, and the remainder when 534 is divided by 13 is 1.)

6. Use the test in question 5 to determine if the number 12,345,600,001,234,563 is divisible by any of 7, 11, or 13.

# 2 Coding Theory

**Read Sections 3.1, 5.7, and 7.1 of *Ecco*.**

The kinds of codes in this chapter are different from a *secret code* in that the intent is to help verify the correct transmission of information rather than to hide it from prying eyes or ears.

The typical scenario when information is being exchanged is that there is a sender, a transmission channel, and a receiver. The sender transmits a string of symbols—usually a string of letters or digits. The message may be corrupted in some way by the transmission channel, (or by some subversive agent) and so the receiver may or may not receive the message that was originally transmitted.

There are two different ways to handle this situation—one way is to simply notify the sender that an error has occurred and to ask for retransmission. This is what occurs in the "Wrong Number" problem in Section 5.7 of *Ecco*. No attempt is made to correct the error—we are only interested in determining whether or not an error has been committed. The solution to the "Wrong Number" problem is an example of an ***error detecting code***.

In the "Campers Problem" (Section 3.1 of *Ecco*), the situation requires not only that we detect an error but also that we correct it. The solution is an example of an ***error correcting code***

## 2.1 Error detection

We begin with what appears to be the simpler problem, that of error detection. Here is Dr. Ecco's "Wrong Number" problem.

**Wrong Number.** The "switch bug" is a glitch in a certain company's telephone apparatus that occasionally transposes adjacent digits among the last five digits of a telephone number. In order to mollify the many people who were receiving telephone calls in error, the telephone company is willing to add an extra digit if a transposition would result in a nonfunctioning number. However, the extra digit itself might take part in a transposition, so Dr. Ecco was consulted about the problem. How can the the extra digit be chosen so that transpositions will be flagged?

The method used to detect errors depends upon the sort of errors that we want to detect. For example, in the "Wrong Number" problem, we are only interested in detecting an error that occurs when two adjacent digits are interchanged. Other errors are possible—someone could unintentionally change one of the digits in the number—but the solution is not necessarily designed to catch such an error. No error detection scheme can find all possible errors.

As Dr. Ecco pointed out when he gave a solution, different solutions are possible. One is as follows: given the five digit number $abcde$, add a sixth digit $x$ so that $b + d + x$ is divisible by 10. The number that is sent is $abcdex$.

To see why this enables us to detect an error, let us consider what happens when $b$ and $c$ are interchanged so that the switch changes the number to $acbdex$. Suppose the altered number passes the test. Then $c + d + x$ is divisible by 10 and as a consequence so is $(b + d + x) - (c + d + x)$ from which it follows that $b - c$ is divisible by 10. Since $b$ and $c$ are both single digits, this can happen only if $b = c$, and in that case there would be no error. To put it another way, if $b$ and $c$ are different, the altered number would not pass the test and we would detect the error.

The extra digit $x$ that is introduced is called a ***check digit***. The other digits are called ***information digits***. The information digits can be freely chosen, but check digits cannot—they are computed using the information digits.

Saying that the check digit $x$ is chosen so that $b + d + x$ is divisible by 10 is the same as saying that $x$ is chosen so that $b + d + x \equiv 0 \pmod{10}$. The proof that transposing $a$ and $b$ will flag an error looks like this when we use modular arithmetic:

For the number $abcdex$ we have

$$b + d + x \equiv 0 \pmod{10}.$$

If the digits $a$ and $b$ are interchanged, the number becomes $bacdex$ and it will be accepted as correct if and only if

$$a + d + x \equiv 0 \pmod{10}.$$

Subtracting one congruence from the other, we get

$$b - a \equiv 0 \pmod{10},$$

or equivalently, $b \equiv a \pmod{10}$, and since both $a$ and $b$ are nonnegative numbers less than 10, we must have $a = b$. In other words, if $a$ and $b$ are not the same, the number $bacdex$ will not pass the test.

There is nothing particularly special about the zero. We could have chosen the check digit $x$ so that $b + d + x \equiv 3 \pmod{10}$, for the only way that $a + d + x$ can also be congruent to 3 is for the digits $a$ and $b$ to be equal.

Almost all of the numbers that you have on plastic cards that you carry with you have at least one check digit, and this is especially true if the number is given by means of a barcode. The purpose of the barcode is enable optical scanning of the number, and the check digit is to ensure that there is no error in transmitting what was scanned. Your driver's license, your credit card, your university ID card all have check digits. Also, the barcodes you see on commercial products generate a 12 or 13 digit number called the UPC (Universal Product Code), or the EAN (European Article Number). The UPC has 12 digits: 11 plus a check digit. The EAN has 13 digits: 12 plus a check digit. (The EAN is also called UPC-13). If you look on the back of the *Ecco* book, you see the ISBN code—this consists of 9 digits plus a check digit (the check digit for an ISBN can be X as well as one of the 10 numerical digits.) If you want to know more about the check digits for credit cards and product codes, see the appendix.

### 2.1.1 What is a code?

In the Wrong Number problem, the solution was to create a special subset of six-digit numbers. These special numbers are *codewords* and the *code* is the collection of all of the codewords. The code does not include all six-digit numbers. The problem the receiver has is to determine whether the six-digit number that he or she receives is one of the code words. One possibility is for the receiver to compare the received word to a list of all the codewords, but this can become quite infeasible if there are a huge number of codewords.

In general, a ***code*** is just a collection of codewords, and a ***codeword*** is a string of digits or letters or other symbols.

This general definition oversimplifies the situation. As mentioned earlier, the usual scenario is that the sender transmits a codeword over some communication channel and the transmitted word is obtained by a receiver. Errors may occur along the way. The communication channel might be noisy and corrupt the message: this is what happened in the Wrong Number problem—the switch board is part of the communication channel and it created random“noise” by occasionally switching digits. For error detection, the code designer has to create a large set of codewords with the property that transmission errors cannot transform any codeword into a different codeword. The choice of codewords has to be governed by the errors that are likely to occur.

### 2.1.2 Binary codes

In everyday usage, the most frequently encountered codes are the ***binary codes***, which consist of strings of 0’s and 1’s, that is, the codewords are binary numbers[1]. In a binary code, all codewords are the same length. A typical setting is that up to a certain number of digits, say $d$, can be corrupted, but the likelihood of more than $d$ being corrupted is very small.

Suppose that in the transmission of a block of digits at most one of the digits may be changed, either from 0 to 1 or vice versa. If all digits are used to represent real information, then an error will cause misinterpretation. We alleviate the problem by trading economy for accuracy, and we use some of the digits solely for the purpose of checking.

For instance, if we are dealing with eight-digit binary numbers, we could use the first seven digits to represent real information. The last digit will be chosen so that the total number of 1’s in the whole block is even. This last digit is called a ***parity check digit***. If any one of the digits, including the last one, is changed, the total number of 1’s will become odd. The receiver will know that something has gone wrong.

[1]If you are not familiar with binary numbers, see the appendix.

This ***parity check code*** is another error detecting code. Although the receiver knows that an error has occurred, it is usually not possible to tell which digit is in error.

Binary codes, however, have the interesting property that if you know that a particular digit is incorrect, then you know what the correct digit must be. Consequently, these are the codes that error correction focusses on.

## 2.2 Error-correction

In the campers problem (Section 3.1 of *Ecco*), the sender of the message is a group of campers, and because some of them may lie, the message may be corrupted by the sender. Let us recall briefly what the Campers problem was:

**The Campers Problem.** A counselor and a group of eight campers are lost while travelling through the woods. They come to an intersection with four roads leading away. Their camp is twenty minutes down one of the roads, but they do not know which one. There is just one hour of daylight left, so the counselor can send groups of campers down the roads for twenty minutes to check if the camp is there, and ask them to return to the intersection. The ones who went down the correct road would be able to report that fact, and this will leave twenty minutes for everyone to get to the camp.

The problem is that two of the campers are not always truthful, and you do not know which two they are. How do you divide the campers into groups so that you can tell for sure what road leads to the camp? The untruthful campers don't always lie—sometimes they may tell the truth.

Class-room discussions about the problem usually lead to a solution similar to Dr. Ecco's (page 151 of *Ecco*). Here is the same solution, but with a different slant, and it serves as a good introduction to the notion of a Hamming distance.

Send campers number 1, 2, and 3 down road A, numbers 4, 5, and 6 down road B, and numbers 7 and 8 down road C. (The counselor goes down road D and we may assume that the camp is not down road D, otherwise there is no problem.)

The counselor asks each camper the following question:

*Is the camp on the road that you went down?*

The campers will answer 0 (= no) or 1 (= yes).

Suppose that all of the campers told the truth. Then the only possible answers would be

$$\begin{aligned}
a &= 111 \mid 000 \mid 00 \quad \text{if the camp is down road A}\\
b &= 000 \mid 111 \mid 00 \quad \text{if the camp is down road B}\\
c &= 000 \mid 000 \mid 11 \quad \text{if the camp is down road C.}
\end{aligned}$$

These answers all form binary numbers with 8 digits. The vertical bars could be omitted — they are inserted just for the convenience of separating the answers according to the roads travelled by the campers.

We have just describe a code whose codewords are the binary numbers $a$, $b$, and $c$, and this code has the property that it enables you to not only detect an error, but also to correct it.

If up to two of the campers lied, then the actual answers would differ from a codeword in at most two places, and this enables us to see at a glance what the legitimate answer is.

For example, suppose the answers form the word $100 \mid 001 \mid 11$. This differs from the first codeword, $a$, in five places, differs from $b$ in five places, and differs from $c$ in two places. We can conclude that in this case the camp is down road C.

## The Mathematical Background

The Hamming distance between any two character strings of the same length is defined to be the number of places in which the components of the strings differ. For example, the Hamming distance between the strings "XYab4" and "XZAb5" is 3 because the two strings differ in three places.

In many applications, the strings are binary codewords, and the codewords that comprise the code all have the same number of digits.

For example, if

$$\begin{aligned}
x &= 000111001, \quad \text{and}\\
y &= 101010101,
\end{aligned}$$

then the Hamming distance between $x$ and $y$ is 5.

Let $d(x, y)$ denote the Hamming distance between $x$ and $y$. It is not too difficult to see that the Hamming distance has the following properties:

1. It is never negative: $d(x, y) \geq 0$
2. $d(x, y) = 0$ if and only if $x = y$.
3. It is symmetric: $d(x, y) = d(y, x)$.
4. It satisfies the ***triangle inequality***: If $x$, $y$, and $z$ are three strings of the same length, then
$$d(x, z) \leq d(x, y) + d(y, z).$$

When a function $d$ satisfies all of the above properties it is called a ***metric*** or a ***distance function***. These are the minimal properties that we expect a distance function to possess.

Returning to the campers problem, the counselor would not be able to correct the answer if it was within a Hamming distance of two from more than one of the code words. The triangle inequality shows why this situation can never arise: The Hamming distances between the codewords are

$$d(a,b) = 6, \quad d(a,c) = 5, \quad \text{and} \quad d(b,c) = 5.$$

If $x$ is the word produced by the answers, and if the Hamming distance from $x$ to two of the codewords $p$ and $q$ is no greater than 2, then by the triangle inequality, we would have

$$d(p,q) \leq d(p,x) + d(x,q) \leq 2 + 2 = 4,$$

a clear contradiction to the fact that every pair of codewords is at least 5 units apart.

So, no matter how the liars answer, we can always correct the answer and find the codeword and discover which road to take. This clearly generalizes:

**Theorem 2.2.1.** *If up to $n$ bits of a binary codeword can be corrupted, and if the distance between every pair of codewords is at least $2n+1$, then the errors can be corrected.*

*The correct word is the codeword that is nearest to the corrupted word.*

(The word **bit** is shorthand for **binary digit**.)

The error correction in Theorem 2.2.1 is called ***nearest neighbour decoding***

The converse is also true:

**Theorem 2.2.2.** *If up to $n$ bits of a codeword can be corrupted, and if the distance between some two of the codewords is smaller than $2n+1$, then there is a corrupted message which cannot be corrected.*

For error detection, we have two similar theorems:

**Theorem 2.2.3.** *If up to $n$ characters of a codeword can be corrupted, and if the distance between every pair of codewords is at least $n+1$, then an error can detected*

The converse is also true:

**Theorem 2.2.4.** *If up to $n$ characters of a codeword can be corrupted, and if the distance between some two of the codewords is smaller than $n+1$, then there is a corrupted message which cannot be detected.*

## More about the campers problem

Remember that one of the things we like to do is to find out whether the solution is optimal. This is the point of Dr. Ecco's next question:

> *Is it possible to solve the problem with 7 campers, again with two liars among them?*

(As before, we are assuming that the counselor asks one and the same question of all campers.)

There are many ways to send seven campers down three different roads. Start by considering one possibility. Suppose for example, the counsellor sends 3 down road A, 2 down B and 2 down C. In this case, the codewords (the truthful answers) are

$$\begin{array}{ll} a = 111 \mid 00 \mid 00 & \text{If the camp is down road A} \\ b = 000 \mid 11 \mid 00 & \text{If the camp is down road B} \\ c = 000 \mid 00 \mid 11 & \text{If the camp is down road C.} \end{array}$$

The distance between the codewords $b$ and $c$ is less than five. If the answer is $x = 000 \mid 10 \mid 10$, then it is within Hamming distance 2 of both $b$ and $c$, so we could not tell which was the correct codeword. So this way will not work.

Suppose there are two roads, say B and C, down which a total of four or fewer campers go. If $b$ and $c$ are the corresponding codewords when the camp is down B or C, then the Hamming distance between $b$ and $c$ is four or less. So what we want to do is show that no matter how we split up the seven campers, there are always two roads down which four or fewer go. Although there seems to be a large number of possibilities for us to to consider, we can show this without a case by case analysis.

Since there are three roads other than the one down which the counselor goes, we have to split the seven campers into two or three groups. (If we sent all the campers down one road, say A, and if the camp were down B or C, we could not deduce which one.)

We claim that no matter how we split 7 people into 3 groups, there will be two groups whose combined size is four or less. To see why the claim is true, we use a proof by contradiction.

Begin by assuming that the totals that go down each pair of roads is greater than 4. Let $p$, $q$, and $r$ be the number of campers that go down roads A, B, and C respectively. Then, we have

$$\begin{array}{ccccccc} p & + & q & & & \geq 5 \\ p & & & + & r & \geq 5 \\ & & q & + & r & \geq 5 \end{array}$$

Adding the three inequalities, we get

$$2(p + q + r) \geq 15. \tag{2.1}$$

On the other hand, since $p+q+r = 7$, we have $2(p+q+r) = 14$ which contradicts inequality (2.1). This shows that the assumption that the totals that go down each pair of roads is greater than 4 must be wrong, and completes the proof.

The next question that Dr. Ecco asks is really an invitation to extend the problem.

*If there are 5 liars what is the minimum number of truth tellers that are needed?*

The actual answer can be as far away as five from the correct answer, so there must be a separation of at least eleven between each pair of correct answers.

We can certainly do it with 17, that is with 12 truth tellers and 5 liars: Send six campers down roads A and B, and send five down road C. The corresponding codewords are:

111111 | 000000 | 00000
000000 | 111111 | 00000
000000 | 000000 | 11111

Using this scheme, there is a Hamming distance of at least eleven between each pair of codewords. Since there are at most five lies, a corrupted answer can differ from a legitimate one in at most five places, so we can find the correct answer by seeing which codeword is the closest to the corrupted one.

Can we do it with fewer than 12 truth tellers?

The answer is no. The reason is that with 16 or fewer campers, there must be two paths down which at most 10 campers travel, and the corresponding legitimate answers would be separated by a Hamming distance of at most 10. (We leave to you the proof that it is not possible for 11 or more campers to go down each pair of roads.) Consequently, there is a corrupted codeword that is within 5 units of both of the legitimate answers, and so we could not correct that corrupted codeword.

### 2.2.1 Binary Codes and single digit error correction

The analysis of the Campers Problem provides an example of a error-correcting binary code, but there are many others, and one of the earliest was the ***Alt*** code, which is more commonly known as the ***triple repetition code***. This code is able to correct any single digit error.

Here's how it works. Suppose we had four information digits that we wish to transmit. The corresponding codeword would be the twelve digit number created by repeating the information digits three times. If a transmission error changes one of the digits, the majority rule decides what the correct four-digit message is.

For example, if the information digits are 1001, then the codeword is 1001 1001 1001 (the spaces are just to make reading easier). If the received word is 1001 1101 1001, the majority rule tells the receiver that the information digits are 1001.

This is not a very efficient method (but it does have the merit of working with a string of characters from any alphabet, not just words from an alphabet consisting of only two symbols).

Of course, this is not a very efficient method. For binary information an improvement is a double repetition code with a parity-check digit. This code, named the ***Rabenstein*** code after the student who invented it, works like this: Suppose the information digits are $a_1a_2a_3a_4$. The corresponding codeword is $a_1a_2a_3a_4a_1a_2a_3a_4b$, where $b$ is chosen so that the number of ones in $a_1a_2a_3a_4b$ is even. If the first four digits of the codeword do not coincide with the second four, then the parity-check digit tells which four are correct.

## 2.3 The Hamming codes

The infamous Baskerhound's first kidnapping caper can be summarized as follows:

**The Puzzle-Mad Kidnapper** (Section 7.1 of *Ecco*). Baskerhound has kidnapped the son of a wealthy heiress and has sent her the following message.

"I am thinking of a number between 1 and 2000. If you can determine what that number is in 15 or fewer questions, we will release your son. Otherwise we will kill him. I will answer each question with a yes or a no. But beware, I may lie once. Also, I will answer your questions only after you have asked all of them."

What are the fifteen questions.

To find a number between 0 and 2000, we can use a binary search. This is the following sequence of questions (assuming that all of the answers are truthful):

Q1. "Is the number between 0 and 1000?"

Q2. If the answer to the first question is "Yes", then ask "Is the number between 0 and 500?" Otherwise, ask "Is the number between 1001 and 2000?"

We continue in this way, cutting the possible numbers in half with each step. The process will take 11 questions.

This is an adaptive solution. A nonadaptive variant is to first ask the person to write the number in binary form (without revealing the number, of course). Every number between

0 and 2047 and be written using eleven binary digits, so that

$$\begin{aligned}00000000000 &= 0\\ 00000000001 &= 1\\ 00000000010 &= 2\\ 00000000011 &= 3\\ &\vdots\\ 11111111111 &= 2047\end{aligned}$$

Then ask the following questions: "Is the first (leftmost) digit a 1?" "Is the second digit a 1?", and so on. The answers can be supplied all at once, and you can easily determine the number.

In the "Puzzle-Mad Kidnapper", Baskerhound may lie once, so the eleven questions would not produce a correctable answer. You can overcome this by asking three questions for each digit ("Is the $n^{\text{th}}$ digit a 1? Is the $n^{\text{th}}$ digit a 1? Is the $n^{\text{th}}$ digit a 1?". Take the majority answer as being truthful. This, of course, is the *triple repetition code* described on page 24. Unfortunately, this would require 33 questions, but the kidnapper only allows fifteen or fewer questions. The Rabenstein code described earlier would enable us to solve the problem with just 23 questions. This is considerably better, but still a far way from 15.

The ultimate solution is to use a Hamming code. To introduce it, we have dug into our past records for a problem that Dr. Ecco solved but which was not reported in his books:

**The 16 letter alphabet.** General Lange requested Ecco's help once more. "I have to communicate some information in a language that has a 16 letter alphabet. The information is not secret, but we are having trouble with the transmission channel. Every so often, the message is corrupted.

"Here is what happens," said the general. "As you might expect, we are encoding each letter using one of the 4-bit binary words from 0000 through 1111. Occasionally, one of the bits gets flipped. For example, when we send 0110 it could be transmitted perfectly, or it could be turned into 1110, 0010, 0100, or 0111. It never happens that more than one bit is changed, but since we need to use all sixteen letters, we find that we sometimes have to repeat a message several times before it is understood. "The question, Dr. Ecco, is whether there is a way around this problem."

Ecco asked "What is the 'bandwidth' of your transmission channel? That is, can you send more than four bits at a time?"

General Lange replied that it is possible to send up to 8 bits at a time. Ecco said that in this case there is indeed a solution.

Recall that in order to be “$n$-error correcting”, a binary code must be such that the distance between codewords is at least $2n+1$, and when this happens, we can use nearest neighbour decoding to correct a received word.

The optimal result for single digit error correction is achieved using ***Hamming codes***, and it is a Hamming code that provides the solution to both the 16-letter alphabet problem and the Puzzle-mad Kidnapper problem. Hamming codes were invented by Richard W. Hamming, in fact, coding theory actually began with Hamming. In 1947 he was working with a mechanical computer that could detect errors, and which brought all work to a crashing halt when such an error was detected. After two successive weekends of having the computer dump his work because of an error, he reports that he said to himself “Damn, if the machine can detect an error, why can’t it locate the position of the error and correct it?”

The problem is that in order to correct a one bit error, the codewords must be separated by a hamming distance of at least 3. Since all 4-bit words need to be used, this is only possible if more bits are added. The following shows how it can be done. Write down the 4-bit word that you want to send. Suppose it is 1010. Leave room for 3 more bits, and above each bit place the letters $a$, $b$, and $c$ as in the following table.

| | | | | | | |
|---|---|---|---|---|---|---|
| $a$ | $a$ | $a$ | | $a$ | | |
| $b$ | $b$ | | $b$ | | $b$ | |
| $c$ | | $c$ | $c$ | | | $c$ |
| 1 | 0 | 1 | 0 | -- | -- | -- |

Each of the seven columns contains a different nonempty subset of $\{a, b, c\}$. Now, place 0 or 1 in the position beneath the singleton $\{a\}$, so that the total number of 1’s in all columns containing $a$ is even. For example, there are two 1’s under the $a$’s to the left of the vertical line, so place 0 under the single $a$ to the right of the vertical line. The remaining bits are chosen in a similar manner. There is one 1 under the $b$’s to the left of the vertical line, so place 1 under the $b$ to the right of the vertical line. Continue in this fashion, and place 0 under the $c$ and you will get the following table:

| | | | | | | |
|---|---|---|---|---|---|---|
| $a$ | $a$ | $a$ | | $a$ | | |
| $b$ | $b$ | | $b$ | | $b$ | |
| $c$ | | $c$ | $c$ | | | $c$ |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 |

This results in the 7-bit word 1010010, and this is one of the codewords that may be transmitted.

Any one of these seven bits might be changed during transmission, but the error can be corrected. Suppose that the receiver gets the corrupted word 1110010. The decoder begins by writing down a similar table with the received word in the bottom line (it is crucially important that sender and receiver have the same order of the columns):

| | | | | | | |
|---|---|---|---|---|---|---|
| $a$ | $a$ | $a$ | | $a$ | | |
| $b$ | $b$ | | $b$ | | $b$ | |
| $c$ | | $c$ | $c$ | | | $c$ |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 |

For each letter, the decoder checks that there is an even number of 1's beneath the columns that contain that letter. For this particular example,

> Below the $a$'s there are 3 ones, so there is an error in a column that contains an $a$.
>
> Below the $b$'s there are 3 ones, so there is an error in a column that contains a $b$.
>
> Below the $c$'s there are 2 ones, so there are no errors in a column containing a $c$.

The decoder can conclude that the error occurs in the column containing $a$ and $b$.

There are also Hamming codes with 15 digits, 31 digits, and in general with $2^k - 1$ digits. These codes have $k$ parity-check digits. For example, the Hamming code with 15 digits has 4 check digits and 11 information digits. The information digits may be freely chosen and the check digits are calculated using the nonempty subsets of the set $\{a, b, c, d\}$.

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $a$ | $a$ | $a$ | $a$ | | $a$ | $a$ | $a$ | | | | $a$ | | | |
| $b$ | $b$ | $b$ | | $b$ | $b$ | | | $b$ | $b$ | | | $b$ | | |
| $c$ | $c$ | | $c$ | $c$ | | $c$ | | $c$ | | $c$ | | | $c$ | |
| $d$ | | $d$ | $d$ | $d$ | | | $d$ | | $d$ | $d$ | | | | $d$ |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | -- | -- | -- | -- |

Each column represents a different non-empty subset of $\{a, b, c, d\}$, the elements being listed in the first four rows. The first eleven digits in the fifth row constitute the intended message. The twelfth digit, under the singleton subset $\{a\}$, is chosen so that the total number of 1's in all columns containing the element $a$ is even. The remaining three digits are chosen in an analogous manner, but in connection with the elements $b$, $c$, and $d$ respectively. When this is done it generates the codeword 100111011010011.

Suppose the message received is as shown in the table below.

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $a$ | $a$ | $a$ | $a$ | | $a$ | $a$ | $a$ | | | | $a$ | | | |
| $b$ | $b$ | $b$ | | $b$ | $b$ | | | $b$ | $b$ | | | $b$ | | |
| $c$ | $c$ | | $c$ | $c$ | | $c$ | | $c$ | | $c$ | | | $c$ | |
| $d$ | | $d$ | $d$ | $d$ | | | $d$ | | $d$ | $d$ | | | | $d$ |
| 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |

The receiver will find out that the total numbers of 1's under all columns containing the elements $a, b, c$ and $d$ are respectively 3, 4, 5 and 5 respectively. Thus the parity check is failed by $a$, $c$, and $d$ but not by $b$. Since there is at most one error, it must have occurred in the column corresponding to the subset $\{a, c, d\}$.

### 2.3.1 The Baskerhound case

The first eleven questions you should ask are the nonadaptive ones we listed on page 26. That is, have Baskerhound convert his number to binary and ask the following: "Is the first (leftmost) digit a 1?" "Is the second digit a 1?", and so on.

Then, questions 12 through 15 are:

12. Are there an odd number of 1's among the digits in positions 1, 2, 3, 4, 6, 7, and 8?
13. Are there an odd number of 1's among the digits in positions 1, 2, 3, 5, 6, 9, and 10?
14. Are there an odd number of 1's among the digits in positions 1, 2, 4, 5, 7, 9, and 11?
15. Are there an odd number of 1's among the digits in positions 1, 3, 4, 5, 8, 10, and 11?

The answers can be supplied all at once, and even if Baskerhound lies once, you can easily determine the number using the Hamming code.

### 2.3.2 The optimality of the Hamming Code

When we say that the Hamming Code is optimal for words of length, say, fifteen, we mean that there is no other single-error-correcting code that has more codewords.

To make this a little clearer, consider a situation with binary codewords of length 6. There are $2^6$, or 64, words of length 6, but not all of them can be codewords. One possible single error correcting code consists of the four words

$$w_1 = 000000, w_2 = 111111, w_3 = 000111, w_4 = 111000. \tag{2.2}$$

This corrects any single digit error because the minimum distance between any two of the words is 3. We cannot add any more words to the code. The reason is as follows: if the Hamming distance between the word $x = abcdef$ and each of $w_1$ and $w_2$ is at least three, then exactly three of the digits must be ones and three of the digits must be zeros. By the pigeon-hole principle, either $\{a, b, c\}$ or $\{d, e, f\}$ contains two of the ones. It follows that either $d(x, w_3) = 2$ or $d(x, w_4) = 2$, so if $x$ were included in the code, it would not be possible to correct all single digit errors. So it looks like this is as good as we can do.

Although we cannot add any more codewords to the set (2.2) there is actually another six-digit code that is better in the sense that it consists of *five* codewords:

$$u_1 = 000000, u_2 = 111111, u_3 = 000111, u_4 = 110001, u_5 = 011010. \tag{2.3}$$

When we say that a code is ***optimal*** for words of a given length, we mean that it is not possible to find another code that contains more words.

Now let us explain why the Hamming code for words of length 15 is optimal. The number of codewords is $2^{11}$, since only the first 11 digits can be freely chosen. (After the 11 are chosen the remaining 4 are uniquely determined.)

Now let us consider any 15 digit binary code that corrects single digit errors. Suppose that there are $n$ codewords. For each codeword, there are exactly 15 other words at Hamming distance 1 away. Let us call them the associates of with that codeword. None of these associates is a codeword itself. Moreover, they cannot be associates of another codeword, otherwise the Hamming distance between the two codewords would be 2. It follows that each codeword together with its 15 associates forms a set of sixteen words, and the sets do not overlap. The union of all of these sets therefore contains $16n$ words, so that $16n \leq 2^{15}$. But this implies that $n \leq 2^{11}$, so no code can contain more codewords than the Hamming code. In other words, the Hamming code is optimal.

## 2.4 Exercises

1. Construct a table of Hamming distances between any pair of the words 0000, 0011, 0101, 0110, 1001, 1010, 1100 and 1111.

2. Correct any error in the following received word as part of a message in the 15-digit Hamming Code:

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $a$ | $a$ | $a$ | $a$ | | $a$ | $a$ | $a$ | | | | $a$ | | | |
| $b$ | $b$ | $b$ | | $b$ | $b$ | | | $b$ | $b$ | | | $b$ | | |
| $c$ | $c$ | | $c$ | $c$ | | $c$ | | $c$ | | $c$ | | | $c$ | |
| $d$ | | $d$ | $d$ | $d$ | | | $d$ | | $d$ | $d$ | | | | $d$ |
| 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 | 0 |

3. A published book is identified by a ten-digit number known as its ISBN. These digits denote the language, the publisher and other data of the book, except for the last one, which is introduced as a check digit. If the first nine digits are *abcdefghi*, then the tenth digit $j$ is chosen so that $10a + 9b + 8c + 7d + 6e + 5f + 4g + 3h + 2i + j \equiv 0 \pmod{11}$. If $j$ has to be 10, an $X$ is used instead.

   (a) Find the tenth digit of an ISBN if the first nine digits are 0-7167-2314.

   (b) Explain why the ISBN will detect any single digit error.

   (c) Will the ISBN detect an error where two adjacent integers are interchanged? Explain.

   (d) Is the ISBN a single-error-correcting code? That is, will it enable the receiver to correct any single digit error? Explain.

## 2.5 Questions

1. In Section 5.7 of *Ecco*, if the switching error always transposes two digits which have exactly one other digit in between, how should a sixth digit be added so that no wrong number can result from a switching error?

2. A person put 35 beans into 3 jars. Show that there must be two jars with a combined total of 23 or fewer beans.

3. Here is a way to solve Problem 3 of Section 7.1 of *Ecco*, even with the answers coming after all the questions have been asked. In the first 11 questions, we ask in turn whether each of the coins is heads. The next 11 questions are the same as the first 11. Find a suitable last question and explain why it works.

4. The following message is received using the 15-digit Hamming Code. Correct any single error that may have arisen during transmission.

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $a$ | $a$ | $a$ | $a$ | | $a$ | $a$ | $a$ | | | | $a$ | | | |
| $b$ | $b$ | $b$ | | $b$ | $b$ | | | $b$ | $b$ | | | $b$ | | |
| $c$ | $c$ | | $c$ | $c$ | | $c$ | | $c$ | | $c$ | | | $c$ | |
| $d$ | | $d$ | $d$ | $d$ | | | $d$ | | $d$ | $d$ | | | | $d$ |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 0 |

5. In Section 7.1 of *Ecco*, what is the minimum number of questions with a "Yes" or "No" response needed to determine the status of 2 coins, if 2 lies are permitted? All questions are tabled before any answers are given, and no hypothetical questions are allowed.

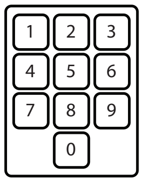
6. In Section 3.1 of *Ecco*, suppose the counsellor has 100 minutes but only three campers, two of whom are liars. However, they will always give the same answer as each other if they go down the same path together. How would the counsellor determine where the campsite is?

7. The following message is received using the 15-digit Hamming Code. Correct the number if it is not correct.

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $a$ | $a$ | $a$ | $a$ | | $a$ | $a$ | $a$ | | | | $a$ | | | |
| $b$ | $b$ | $b$ | | $b$ | $b$ | | | $b$ | $b$ | | | $b$ | | |
| $c$ | $c$ | | $c$ | $c$ | | $c$ | | $c$ | | $c$ | | | $c$ | |
| $d$ | | $d$ | $d$ | $d$ | | | $d$ | | $d$ | $d$ | | | | $d$ |
| 0 | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 0 |

8. A Coffee Club with 8 members meets every Monday, Wednesday and Friday at *Café Rendevous.* At each meeting, 4 members sit at a table indoors and the other 4 at a table outdoors. Is it possible, in the course of a week, for each member to be indoors at least once and outdoors at least once, and be at the same table at least once with each of the other 7 members?

9. A common error in dialing on a standard telephone keypad is to punch in a digit adjacent to the intended one. So, on a standard keypad, 4 could be erroneously entered as 1, 5, or 7 (but not as 2 or 8).



   (a) Assume that the numbers are 5 digits *abcde*, which may be freely chosen, and a sixth check digit $f$ is appended according to scheme used by Dr. Ecco in the Wrong-Number problem (problem 5.7). How good is this at detecting the error described above? (Explain.)

   (b) Another solution was to add the sixth digit $f$ so that $b + d + f \equiv 0 \pmod{10}$. How good is this method? (Explain.)

10. There are two boxes, A and B, belonging to the queen of a small island kingdom. In each box the court jester has placed up to two coins, but not necessarily the same number in each, and perhaps none in some of the boxes. Your task is to find out what is in each box by asking questions of the court jester. Your may only ask questions that have a single digit answer—for example, you could ask "How many coins are there in boxes A and B combined?" The court jester is allowed to lie once, and he will not answer at all if the correct answer is not a single digit. A variation of the triple repetition code will provide an adaptive solution using five questions. Provide these questions.

11. A certain city has 10 million households, each with a single telephone. The telephone numbers run from 0000000 to 9999999. The automated switchboard often transposes two adjacent digits of the number dialed, resulting in a call to a wrong number when these digits are different. Some annoyed customers cancel telephone service. The company wants to devise a scheme where a switchboard error will only result in a call to an inoperative number. Using the schemes discussed for the solution to Dr. Ecco's switch bug problem, where the last digit is used as a check digit, the result would be that only 1 million telephones had a valid number. Devise a scheme where considerably more than 1 million telephones could still have a valid number.

12. In section 2.3.1 the last four questions could have been

    12. Did you lie in any of your answers to questions 1, 2, 3, 4, 6, 7, or 8?
    13. Did you lie in any of your answers to questions 1, 2, 3, 5, 6, 9, or 10?
    14. Did you lie in any of your answers to questions 1, 2, 4, 5, 7, 9, or 11?
    15. Did you lie in any of your answers to questions 1, 3, 4, 5, 8, 10, or 11?

    Explain how you can tell which answer is a lie.

# 3 Cryptography

**Read Sections 4.3, 5.5, and 6.3 of *Ecco***

## 3.1 Simple ciphers

Secret codes seem to be universally popular, among children as well as governments, for fun or for reasons that may be deadly.

Usually, we have a sender-receiver team on one side and a kibitzer on the other. The sender wishes to communicate a message to the receiver, but she is worried that it may be intercepted by the kibitzer. So she agrees with the receiver on a pair of functions $E$ and $D$, called the **encoding** and **decoding** functions, with the property that $D(E(x)) = x$ for any message $x$. Applying the encoding function $E$, the sender sends the encoded message $E(x)$. If the kibitzer intercepts this, all he gets is gibberish. However, the receiver can apply the decoding function $D$ and recover the original message.

The task of the sender-receiver team is to choose encoding and decoding functions which are relatively easy to use and to remember but difficult for the kibitzer to determine. The task of the kibitzer is to try to deduce the decoding function from the samples of coded messages he has intercepted.

### 3.1.1 Caesar's Code

We shall confine our attention to codes in which the encoded message is also a text message. A famous example is ***Caesar's Code***, in which each letter is shifted three places down the alphabetical order. Thus 'A' becomes 'D', 'B' becomes 'E', and so on. At the end of the alphabet, we wrap things around so that 'X', 'Y' and 'Z' become 'A', 'B' and 'C' respectively. The message is decoded by shifting backwards three places. For example, if the received message is 'SUB', shifting back yields 'PRY'.

At this point we should mention that in cryptography, the unencoded message is called the ***plaintext***, and the encoded message is called the ***ciphertext***, so for the previous example 'PRY' is the plaintext and 'SUB' is the ciphertext.

To put Caesar's Code into mathematical language, we label the letters from A to Z with the integers 0 through 25, as shown below:

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | B | C | D | E | F | G | H | I | J | K | L | M |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |

Then the labels are transformed using the encoding function $E(x) = x + 3 \pmod{26}$. More accurately, $E(x)$ is the least nonnegative residue in the residue class $[x + 3]$ (see page 9).

For example, P, R and Y are labelled with 15, 17, and 24 respectively, and $E(15) = 18 = \mathrm{S}$, $E(17) = 20 = \mathrm{U}$ and $E(24) = 1 = \mathrm{B}$. So the encoded message is "SUB".

When the receiver gets the message "SUB", she changes the letters back to the numbers 18, 20 and 1 and applies the decoding function obtained from $D(x) \equiv x - 3 \pmod{26}$. She gets 15, 17, and 24. (The least nonnegative residue in the equivalence class $[-2]$ is 24, which explains why $D(1) = 24$).

This illustrates a very typical process: the characters of the plaintext are converted into numerical values, the encoding function is applied to the numerical values, and the results are converted into their alphabetic counterparts to produce the ciphertext. The process is reversed to decode the ciphertext.

Caesar's Code is certainly a very simple cipher to use, but it is not very difficult to crack. The preservation of the cyclic order of the letters leaves too much structure for the kibitzer. If he receives enough encoded messages, he will very quickly determine the decoding function. Even if the sender-receiver team replaces 3 by some other number as the shift, the kibitzer has a relatively easy time to decrypt an intercepted message if he guesses that such a ***generalized Caesar's Code*** is used, that is, if the encoding function is $E(x) = x + k \pmod{26}$.

For example, suppose the intercepted message is "MKOCKB". If the kibitzer knows that a generalized Caesar's code is being used, he prepares a table with the letters of MKOCKB in the top row, and successive letters of the alphabet in the rows below.

Scanning row by row, the only one which makes sense is "CAESAR". Hence that must have been the original message.

| **M** | **K** | **O** | **C** | **K** | **B** |
|---|---|---|---|---|---|
| N | L | P | D | L | C |
| O | M | Q | E | M | D |
| P | N | R | F | N | E |
| Q | O | S | G | O | F |
| R | P | T | H | P | G |
| S | Q | U | I | Q | H |
| T | R | V | J | R | I |
| U | S | W | K | S | J |
| V | T | X | L | T | K |
| W | U | Y | M | U | L |
| X | V | Z | N | V | M |
| Y | W | A | O | W | N |
| Z | X | B | P | X | O |
| A | Y | C | Q | Y | P |
| B | Z | D | R | Z | Q |
| C | A | E | S | A | R |
| D | B | F | T | B | S |
| E | C | G | U | C | T |
| F | D | H | V | D | U |
| G | E | I | W | E | V |
| H | F | J | X | F | W |
| I | G | K | Y | G | X |
| J | H | L | Z | H | Y |
| K | I | M | A | I | Z |
| L | J | N | B | J | A |

### 3.1.2 Linear Codes

To eliminate this problem, the sender-receiver team may modify Caesar's Code by replacing the encoding function $E(x) \equiv x + 3 \pmod{26}$ with one of the form

$$E(x) \equiv ax + b \pmod{26}.$$

Such a code is called a ***linear code***. For example, suppose $a = 5$ and $b = 3$ in a linear code, and the plaintext is "HIDE". The sender will perform the following calculations modulo 26:

$$5 \cdot 7 + 3 = 38 \equiv 12,\ 5 \cdot 8 + 3 = 43 \equiv 17,\ 5 \cdot 3 + 3 = 18,\ \text{and } 5 \cdot 4 + 3 = 23.$$

Hence the ciphertext is "MRSX".

How will the receiver decode this message? Take the first letter "M". She converts it to 12. Now she is looking for an integer $x$ that satisfies

$$5x + 3 \equiv 12 \pmod{26}.$$

The first step in solving this congruence is easy—she subtracts 3 from both sides:

$$5x \equiv 9 \pmod{26}. \tag{3.1}$$

What she wants to do now is to find an integer $x$ that satisfies the congruence (3.1). Now comes the hard part, because in ordinary arithmetic 9 is not divisible by 5. However, this is modular arithmetic, and any integer in a congruence can be replaced by another integer from the same congruence class. In modulo-26 arithmetic, the equivalence class [9] consists of the numbers that differ from 9 by multiples of 26, that is

$$[9] = \{\ldots, 9 - 2 \cdot 26, 9 - 26, 9, 9 + 26, 9 + 2 \cdot 26, \ldots\}.$$

So she can replace the 9 by any one of these numbers, in particular by 9+26 or 35:

$$5x \equiv 35 \pmod{26},$$

and she can now divide by 5 to get $x = 7$, and "M" translates back into "H" as desired.

She can continue in this way letter by letter, but what she needs is a more efficient way. It is possible to prepare a decoding table, but it is not in the team's interest to have it lying around. Of course, it can be reconstructed every time it is used, but that is cumbersome too. What we need is a good way of performing division in modular arithmetic. This was a topic that we touched upon in Chapter 1, and we now come back to it. We begin with an example.

**Abraham and Norma** or Abe and Nor as they are called by their chums, worked for the now defunct Undermine Investments. Both are known for their savvy, but there is a rumor that one of them had embezzled company funds to the tune of 8 million dollars, which contributed greatly to the demise of the company.

Both have applied to your firm for the vacated precious metals fund manager position. You have hired a private investigator who will communicate with you using the linear code $E(x) \equiv 6x + 5 \pmod{26}$. He will send you a three letter word that is the name of the person whom you should hire. The private investigator has found out that Abraham is the embezzler, so he advises you to hire Norma and sends you the message "FLD" which is what he got when he applied the encoding function to "NOR". (Check this.)

You now begin the decoding process. The numbers corresponding to F, L, and D are 5, 11, and 3. You undo the linear code $6x + 5$ using modular arithmetic as follows: First you subtract 5 from each of the numbers, obtaining 0, 6, and 24, (24, because $24 = (3 - 5) + 26$). Then you divide by 6 to get 0, 1, and 4. These are the letters A, B, and E, and you end up hiring the wrong person. What happened?

### 3.1.3 Finding the decoding function for a linear code

Suppose we are trying to solve the congruence $3x \equiv 13 \pmod{19}$. By a solution we mean the set of all integers $x$ that satisfy the congruence. One integer that satisfies it is $x = -2$, and it follows that the congruence will also be satisfied by every member of the equivalence class $[-2]$. The congruence has the nice property that *all* of its solutions belong to the class $[-2]$. The ABE and NOR situation arose because the solutions for $6x + 5 \equiv 5 \pmod{26}$ do *not* all belong to the same class—the integers in both $[0]$ and $[13]$ satisfy the congruence, and so there is no decoding function.

If the private investigator had used the encoding function

$$E(x) \equiv 7x + 6 \pmod{26}, \tag{3.2}$$

the solutions would all belong to the same class because 7 and 26 are relatively prime. In this case there is a decoding function, and this is how to find it: Replace the $E(x)$ by $y$ in congruence getting

$$y \equiv 7x + 6 \pmod{26}$$

Now solve for $x$ in terms of y. The first step is to subtract the constant 6 from the congruence getting

$$y - 6 \equiv 7x \pmod{26}.$$

The second step is to "divide by 7", which, in modular arithmetic means "multiply by the inverse of 7". In modulo-26 arithmetic, the numbers 7 and 15 are inverses of each other

$(7 \cdot 15 \equiv 1 \pmod{26})$, so multiplying by 15 we get $15y - 90 \equiv x \pmod{26}$ or, equivalently

$$15y + 14 \equiv x \pmod{26},$$

giving us the decoding function $D(y) \equiv 15y + 14 \pmod{26}$.

In modulo-26 arithmetic, we can only divide by 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23 and 25. In other words, these are the only integers (along with their congruent compatriots) that have inverses.

## Finding the inverse

There are several ways of finding the inverse of a given integer. One was is to search for non-prime numbers in the class [1]. Recall that in modulo-26 arithmetic,

$$[1] = \{1, 27, 53, 79, 105, 131, 157, 183, 209, \ldots\}$$

The non-primes are 1, 27, 105, 209, and so on. Factorization of these shows that each of $1 = 1 \cdot 1$, $27 = 3 \cdot 9$, $105 = 3 \cdot 5 \cdot 7$, $209 = 11 \cdot 19$, and so on. So, 1 is its own inverse, 3 and 9 are inverses of each other, 15 and 7 are inverses, 11 and 19 are inverses, and so on.

An alternate way of finding an inverse of a particular number is to test each multiple. For example, to find the inverse of 15, we test $15 \cdot 3$, $15 \cdot 5$, $15 \cdot 7$, $15 \cdot 9$ and so on until we obtain an integer that is congruent to 1. We do not need to test $15 \cdot 2$, $15 \cdot 4$, because 2, 4, *etc.*, do not have inverses in modulo-26 arithmetic.

Both these methods work well for small moduli like 26. However, it is not always easy to factor large integers, and for large moduli it is better to use the ***Euclidean Algorithm***. The method simultaneously tells us whether a given integer has an inverse and what that inverse is.

**Example 3.1.1.** *Find the inverse of 7 in modulo-26 arithmetic.*

*Solution.* The Euclidean Algorithm is usually thought of as a procedure for finding the greatest common divisor (gcd) of two integers, and that's how we begin. By repeatedly applying the division algorithm we first find $\gcd(26, 7)$.

At each point in the algorithm, we have an equation of the form

$$m = q \cdot n + r,$$

which is obtained when $m$ is divided by $n$ giving a remainder $r$. Now if $m$ and $n$ are divisible by $d$, then so are $n$ and $r$, and conversely, so $\gcd(m, n) = \gcd(n, r)$. The next step is to divide $n$ by $r$ to get a new remainder $t$:

$$n = s \cdot r + t,$$

so then $\gcd(n,r) = \gcd(r,t)$. Continue in this fashion until the remained is zero:

$$26 = 3 \cdot 7 + 5, \tag{3.3}$$
$$7 = 1 \cdot 5 + 2. \tag{3.4}$$
$$5 = 2 \cdot 2 + 1 \tag{3.5}$$
$$2 = 2 \cdot 1 + 0. \tag{3.6}$$

The last nonzero remainder is the gcd, so at this point we know that $\gcd(26,7) = 1$, that is, 26 and 7 are relatively prime. This tells us that 7 has an inverse modulo 26.

We now back-step through the algorithm to find integers $p$ and $q$ such that $p \cdot 26 + q \cdot 7 = 1$. (This would mean that $q \cdot 7$ is in the residue class [1], so $q$ and 7 are inverses modulo 26.)

Before beginning, you might find it more convenient to rearrange equations (3.3) through (3.5) to isolate the remainders as follows:

$$26 - 3 \cdot 7 = 5, \tag{3.7}$$
$$7 - 1 \cdot 5 = 2, \tag{3.8}$$
$$5 - 2 \cdot 2 = 1, \tag{3.9}$$

Start the backstepping process with equation (3.9) that is

$$5 - 2 \cdot 2 = 1.$$

We replace the 2 by its value from equation (3.8), that is, we replace the 2 by $7 - 1 \cdot 5$ getting

$$5 - 2 \cdot (7 - 1 \cdot 5) = 1, \quad \text{that is,} \quad 3 \cdot 5 - 2 \cdot 7 = 1.$$

Next, we replace the 5 by its value in equation (3.7), getting

$$3 \cdot (26 - 3 \cdot 7) - 2 \cdot 7 = 1,$$

which, upon collecting terms, becomes

$$3 \cdot 26 - 11 \cdot 7 = 1.$$

Thus $p = 3$ and $q = -11$, so the inverse of 7 is $-11$ (or 15 since $-11$ and 15 are congruent modulo 26). □

When the Euclidean ALgorithm is followed by back-stepping in this manner, the process is often called the ***Extended Euclidean Algorithm***.

### 3.1.4 Mono-alphabetic codes

The method used to tackle the generalized Caesar's Code is not applicable to the linear codes. However, as long as the same letter is always replaced in the same way and different letters are replaced in different ways, the kibitzer can resort to statistical analysis. Such a code is called a **mono-alphabetic code**. *Note that a mono-alphabetic code is not necessarily a linear code. The encoding function is not necessarily of the form* $E(x) \equiv ax + b \pmod{26}$.

Large random samples of English prose have been examined for the compilation of statistical data. It is generally agreed that the letter E appears far more often than any others. The next two on the frequency hierarchy are probably A and T, though I, N, O, R and S are not far behind. For very long passages of English text, the following percentages have been observed:

| A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 8.04 | 1.54 | 3.06 | 3.99 | 12.51 | 2.30 | 1.96 | 5.49 | 7.26 | 0.16 | 0.67 | 4.14 | 2.53 |
| N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| 7.09 | 7.60 | 2.00 | 0.11 | 6.12 | 6.54 | 9.25, | 2.71 | 0.99 | 1.92 | 0.19 | 1.73 | 0.09 |

In order of frequencies:

| 12.51 | 9.25 | 8.04 | 7.60 | 7.26 | 7.09 | 6.54 | 6.12 | 5.49 | 4.14 | 3.99 | 3.06 | 2.71 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| E | T | A | O | I | N | S | R | H | L | D | C | U |
| 2.53 | 2.30 | 2.00 | 1.96 | 1.92 | 1.73 | 1.54 | 0.99 | 0.67 | 0.19 | 0.16 | 0.11 | 0.09 |
| M | F | P | G | W | Y | B | V | K | X | J | Q | Z |

Other passages may give slightly different frequencies, but the point is that most long passages are close to these.

For example, suppose the kibitzer intercepted the following message: `UMX LXPPDHX RP QVU DP MDKS UV SXNKTAU DP UMX DEXKDHX AXKPVQ LDT UMRQB.`

The letter count for the most frequently occurring letters are

`X` (8 times), `D` (7 times), `U` and `P` (6 times), `M` and `K` (4 times).

Assuming that the code is mono-alphabetic, it is quite probable that the letter which appears most often, namely `X`, stands for `e`, and the one which appears second most often, namely `D`, stands for `t` or `a`. The decoding is likely to be

```
UMX LXPPDHX RP QVU DP MDKS UV SXNKTAU DP UMX DEXKDHX AXKPVQ LDT UMRQB.
--e -e--t-e -- --- t- -t-- -- -e----- t- --e t-e-t-e -e---- -t- -----
```

or else

```
UMX LXPPDHX RP QVU DP MDKS UV SXNKTAU DP UMX DEXKDHX AXKPVQ LDT UMRQB
--e -e--a-e -- --- a- -a-- -- -e----- a- --e a-e-a-e -e---- -a- -----
```

When you are deciphering a message, it may help to keep the plaintext and ciphertext separate by using different cases for each. This is the convention used in Simon Singh's book[1], where the plaintext is in lowercase and the ciphertext is in uppercase.

[1] *The Code Book: the Science of Secrecy from Ancient Egypt to Quantum Cryptography*, Anchor Books, New York, 2000

After a partial decryption, the kibitzer can use the form of the words as a basis for further exploration. Of the two options above, the second seems more likely because the first word is probably `the` (which it couldn't be in the first option because `t` is already used).

Assuming that the first word is indeed `the`, replacing the `U` by `t` and the `M` by `h` throughout we get

```
UMX LXPPDHX RP QVU DP MDKS UV SXNKTAU DP UMX DEXKDHX AXKPVQ LDT UMRQB
the -e--a-e -- --t a- ha-- t- -e----t a- the a-e-a-e -e---- -a- th---
```

The appearance of `t-` for `UV` suggests that `V` is `o`, so we now get

```
UMX LXPPDHX RP QVU DP MDKS UV SXNKTAU DP UMX DEXKDHX AXKPVQ LDT UMRQB
the -e--a-e -- -ot a- ha-- to -e----t a- the a-e-a-e -e--o- -a- th---
```

Continuing in this way, it is not hard to complete the decryption.

On the other hand, if it is if it is known that the code is a linear code, then as soon as we have figured out two of the letters, namely that `e` (= 4) encodes into `X` (= 23) and that `a` (= 0) encodes into `D` (= 3), we have

$$a \cdot 4 + b \equiv 23 \pmod{26},$$

and

$$a \cdot 0 + b \equiv 3 \pmod{26}.$$

It follows that $b = 3$ and $a = 5$.

The fact that spaces are preserved from the plaintext to the ciphertext gives too many clues to the kibitzer. In practice, one omits the spaces altogether and breaks the message at every fifth or sixth letter, so the receiver gets:

```
UMXLX PPDHX RPQVU DPMDK SUVSX NKTAU DPUMX DEXKD HXAXK PVQLD TUMRQ B
```

This is a little bit harder to crack, but a frequency analysis still will work.

### 3.1.5 Poly-alphabetic codes

To foil statistical analysis, the sender-receiver team may use a ***poly-alphabetic*** code. Here, a letter is not always substituted in the same way. A poly-alphabetic code often employs a ***key phrase*** known only to the sender and the receiver. For example, the team may have chosen `SECRET` as their key. When needed, the table below right can easily be reconstructed.

| a | S | E | C | R | E | T |
|---|---|---|---|---|---|---|
| b | T | F | D | S | F | U |
| c | U | G | E | T | G | V |
| d | V | H | F | U | H | W |
| e | W | I | G | V | I | X |
| f | X | J | H | W | J | Y |
| g | Y | K | I | X | K | Z |
| h | Z | L | J | Y | L | A |
| i | A | M | K | Z | M | B |
| j | B | N | L | A | N | C |
| k | C | O | M | B | O | D |
| l | D | P | N | C | P | E |
| m | E | Q | O | D | Q | F |
| n | F | R | P | E | R | G |
| o | G | S | Q | F | S | H |
| p | H | T | R | G | T | I |
| q | I | U | S | H | U | J |
| r | J | V | T | I | V | K |
| s | K | W | U | J | W | L |
| t | L | X | V | K | X | M |
| u | M | Y | W | L | Y | N |
| v | N | Z | X | M | Z | O |
| w | O | A | Y | N | A | P |
| x | P | B | Z | O | B | Q |
| y | Q | C | A | P | C | R |
| z | R | D | B | Q | D | S |

The guide column consists of the letters in their natural order. The subsequent columns are cyclic permutations, with the first letter being the corresponding letter in the key phrase. If the plaintext message is `tar`, the ciphertext is `LET`. These three letters are obtained by reading the `t` row and the first column, the `a` row and the second column, and the `r` row and the third column. For longer messages, cycle through the key phrase as many times as is necessary.

However, if the key phrase is shorter than than the message, then frequency analysis can still be used. For example, suppose the key phrase is SECRET, and the plaintext has several hundred characters. Every sixth letter belongs to the same mono-alphabetic code, and if the kibitzer knows this she can easily crack it. Even if she doesn't know that the key phrase has six characters, she can assume it has 2 characters and if this doesn't work, then 3 characters, and so on. With an ordinary desktop computer, it would fall readily to a statistical analysis.

#### *Mono-graphic and poly-graphic codes*

All the codes described so far are examples of ***mono-graphic*** codes, in which each letter in the message is treated as an entity by the encoding function. In a ***poly-graphic*** code, groups of letters are transformed as units. They are even more difficult to crack. We will not go into details here.

### 3.1.6 Public key cryptography

Suppose many people want to communicate with one another in secret codes. Then there has to be a different pair of encoding and decoding functions for each pair of them. This would be extremely cumbersome. Imagine if you have a direct-dial phone for each of your friends. You will not be able to get anywhere because the phones will be in the way.

What we need is a system of secret codes similar to our phone system. Each person will have his or her own pair of encoding and decoding functions $E$ and $D$. The encoding functions of everybody are published in a directory similar to our phone book. If people want to send you a secret message, they will just look up your encoding function, which is public knowledge. Thus this system is called a ***public key code***.

We have now achieved efficient communication, but what about secrecy? The decoding functions of course are known only to their owners, but wouldn't the knowledge of the encoding function let the cat out of the bag? For instance, if we know the encoding function $E(x) \equiv x + 3 \pmod{26}$ of Caesar's Code, we can easily deduce that the decoding function must be $D(x) \equiv x - 3 \pmod{26}$.

To make public key codes possible, the encoding function $E$ and the decoding function $D$ must have, in addition to $D(E(x)) = x$ for all $x$, the unusual property that knowing what $E$ is does not help anyone to determine what $D$ is. So you see that while the concept of public key codes is relatively simple, the technical implementation is non-trivial.

What we need are functions such that their inverse functions cannot be determined without inside knowledge. Such functions are called ***trapdoor functions***. It is very counter-intuitive that they can even exist. We will not carry out explicit constructions, but only outline what may lead to such a function.

Consider modular arithmetic. Suppose the function $E$ is to find the remainder when a given integer is divided by 26. Then $E(105) = 1$. However, there is no inverse function $D$ since $D(1)$ can be any of infinitely many possibilities. However, it may be possible to construct a composite encoding function involving several remainder functions with different moduli.[2]

It is often stated that "subtraction" is the inverse operation of "addition". Strictly speaking, this is not correct. In Caesar's Code, "subtracting 3" is the inverse operation of "adding 3". However, if we apply the operation of "addition" to 2 and 3 and get 5, the inverse operation should produce both 2 and 3 from 5. In other words, it is not "subtraction" but "partition".

Of course, as in the example using modular arithmetic, the answer is not unique. Even if we restrict the summands to positive integers, we still can have 5=1+4. However, it is possible to use partitions as the basis for constructing trapdoor codes known as ***knapsack codes***, the first of which was constructed by Hellman and Diffie who introduced the concept of public key codes.

In a similar manner, the inverse operation of "multiplication" is "factorization". If we restrict ourselves to prime factorization, then the answer is in fact unique, although the process of recovering the prime factors is general extremely difficulty. Based on this idea, Rivest, Shamir and Adleman constructed the class of trapdoor codes known by their initials as **RSA codes**.

As seen in Section 5.5 of *Ecco*, the use of a public key code allows the signing of contracts by mail. The secret decoding functions of the parties serve as their signatures.

[2]If you want to look further into this, read about the Chinese Remainder Theorem in a Number Theory text.

## 3.2 The Couriers Problem

In Section 6.3 of *Ecco*, the encryption is of a different nature: In order to decrypt the code, the kibitzer must have all parts of the message. The idea here is to split the code into several parts, ship several copies and hope that the kibitzer does not get a collection of all parts.

In particular, the Director has a new top secret code, which has five parts, A, B, C, D, and E, which has to be delivered across enemy lines. If the enemy gets all five, "we will be in big trouble". If the enemy gets only four parts, there is no danger, but four parts alone would also be useless to our agent. The Director is willing to use up to eight couriers, and they are sure that no more than two can be caught. If the enemy catches a courier, the enemy gets possession of all of the parts he carries.

The question is to find a distribution of copies of the five parts with eight couriers so that no two carry all five parts and every six do.

One way to solve a problem is to simplify the conditions so that you get a better feel for it. Here is how this might be done for Problem 6.3 of *Ecco*.

**Behind Enemy lines.** If our allies had a particular weapon, we would win the war. We have many copies of the weapon, but to get it to our allies the weapon has to be transported across enemy lines. We don't want the weapon to fall into enemy hands where it could be used against us.

We will disassemble the weapon into several parts. Without all of the parts, the weapon cannot function. So the idea is to send packages of parts across enemy lines. It is known that the enemy can capture *at most one* package. If each copy of the weapon is disassembled into 3 parts, A, B, and C, how many packages are required to get a copy across the lines without allowing the enemy to capture a complete weapon?

Begin by noting that no package can contain a complete weapon, and we must guarantee that if we send $n$ packages, then every $n-1$ packages have enough parts to make a complete weapon.

So one solution is: A, A, B, B, C, C. Is there a more efficient solution, that is, *can we do it with fewer packages?*

The answer is yes: AB, BC, AC.

*Can we do it with fewer than three packages?* No, because each package could have at most two of the three parts, and only one package might get through.

Here's a new problem: *Suppose that the enemy could capture up to two packages. If we split each weapon into* 3 *parts, how many packages do we have to send?*

We can do it with nine (A, A, A, B, B, B, C, C, C), but not with eight. To do it with eight, since we have to send three copies of each part in different packages, one package would contain two different parts. Suppose one package contained A and B. Since we also must ship copies of part C, a different package must contain a copy of C, and together with the first package we would have two packages that contain A, B, and C.

Now we start to expand this new problem.

*Suppose we disassemble the weapon into* 4 *parts,* A, B, C *and* D. *If the enemy can capture at most two packages, can we accomplish our objective with fewer than* 9 *packages?*

Begin by asking "What is the maximum number of different parts in a package?"

The answer is that no package can contain 3 parts, for then no other pack could contain the remaining part without the possibility of the enemy capturing all four parts.

We must send at least three copies of each of the four parts, so the total number of parts we can send is at least twelve. So we could do it by sending twelve packages each with one part. And since there can be at most two parts in any package, we must send at least 12/2, or six, packages.

Let's first try for eight packages. We observe that

1. if one package contains A and B, no other package can contain C and D (for then the enemy could get all 4 parts by intercepting two of the packages).

2. With eight packages, at least four must contain 2 parts (because at least 12 parts are shipped in all).

3. We show that there is no eight-package solution where three of the packages are AB, AC, and AD.

   The reason is that there must be at least one other package with two parts. By (1), it cannot be CD, BD, or BC, so it must be AB, AC, or AD. We may as well assume that it is AB. This doesn't look so good because now we have four copies of A, and we still have to pack 2 of B, 3 of C, and 3 of D into the remaining four packages. But the C's and D's together are 6 parts, and so two of the remaining four packages will be CD and CD. Oops! Then the enemy could get all four parts by capturing an AB and a CD package. This shows is that if we send eight packages, three of them cannot be AB, AC, and AD.

This suggests we try AB, AB, AC, BC as a possible combination, and in fact it works:

| | |
|---|---|
| package #1: | AB |
| package #2: | AB |
| package #3: | AC |
| package #4: | BC |
| package #5: | C |
| package #6: | D |
| package #7: | D |
| package #8: | D |

To check that it works, note that a copy of each part is in three different packages, so at least one copy of each part must get through. On the other hand, no two packages together contain all three parts: Given two of the above 8 packages, if one of them contains only one part, the two together cannot contain 4 parts. If both packages each contain two parts, then part D will be missing from the combination.

*Can we do it with seven packages and* 4 *parts, still assuming that at most two packages may be captured?*

Using the same reasoning as before, since 12 parts in all must be shipped, five of the seven packages must contain 2 parts each. No two of these five can contain all four parts between them. We now show that between them the five cannot contain copies of all 4 parts.

Here's why: We can assume that one of the five is AB. If both C and D occur among the remaining four packages, they cannot occur together, so we can assume that one of the remaining four packages is AC. Now there is a problem. The package that contains D cannot be BD because of AC; it cannot be CD because of AB, and it cannot be AD because we would have the three packages AB, AC, and AD. By observation number 3 on the previous page, there is no eight-package solution containing packages AB, AC, and AD, so there cannot be a seven-package solution.

So among them, the five packages fail to contain one of the parts, say D. But this means that there are only two other packages, and so we could not ship three copies of D, and part D might not get through. This shows that there is no seven-package solution.

The next step is to consider what happens if we break the weapon into five parts—and this, as you recognize, is really the couriers problem, and it can be solved using these ideas.

On the other hand, on page 118 of *Ecco*, we are asked for a solution with only five couriers. This is the minimum to guarantee that a majority of them are not captured. Into how many parts do we have to divide the message then?

We can reason as follows. Let the couriers be numbered 1, 2, 3, 4 and 5. Now between 1 and 2 there must be some part which neither has. Call it A. Since there are three copies of A, all of 3, 4 and 5 are carrying one. Similarly, 1 and 3 both miss some part, and this must be a new part B. Since there are ten possible pairings among the couriers, we have to divide the message into ten parts. A possible scheme is shown in the following table.

| Captured | 1 | 1 | 1 | 1 | 2 | 2 | 2 | 3 | 3 | 4 | C |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Couriers | 2 | 3 | 4 | 5 | 3 | 4 | 5 | 4 | 5 | 5 | # |
| Parts |  |  |  |  | E | F | G | H | I | J | 1 |
| carried |  | B | C | D |  |  |  | H | I | J | 2 |
| by each | A |  | C | D |  | F | G |  |  | J | 3 |
| of the | A | B |  | D | E |  | G |  | I |  | 4 |
| Couriers | A | B | C |  | E | F |  | H |  |  | 5 |

In the table, the last five rows indicate what components are carried by couriers 1 through 5. The middle ten columns correspond to components A through J. The column which corresponds to component A is headed by the captured couriers 1 and 2, so the A components are removed from the rows for couriers 1 and 2, while the A components remain in the rows for couriers 3, 4, and 5.

## 3.3 Exercises

1. If a mono-alphabetic code is applied to the following passage, explain why it may be difficult to decrypt:

    The Sin of Omission

    > Around midnight, a sly-looking man slips into a luxury city building. A woman occupant, watching his actions from a fourth-floor window, grows suspicious and dials 911 for a patrol car. This lady complains, "A man in a brown suit, with shaggy hair, a slight build, and a criminal air is prowling through my lobby."
    >
    > Fairly soon two young cops, Smith and Jarvis, pull up. Looking for an unknown vagrant, Smith spots Jim Oats walking out a front door. Oats, a minor burglar, is bold as brass, arrogant, and calm. Smith grabs him by his collar.
    >
    > "O.K., Oats," snarls Smith, "what brings you to this location?"
    >
    > Fixing his captor with a chilly look and frosty indignation, Oats quips, "I can go on a short stroll. Lift your filthy hands off my shirt. I'm not guilty of anything."
    >
    > Smith drops his hands limply. This haughty air is too much for him to swallow. Angrily Smith says, "What a story. I'm nobody's fool, you punk. I just wish I could put you back in jail, but I can't obtain any proof against you. You know all about why I'm at this building—a station log full of burglary, arson, and muggings."
    >
    > "Now, now," Oats laughs, "think of my rights. How can you talk this way?" Smith's probing hands start to frisk Oats for guns, narcotics, anything unlawful or contraband. Nothing shows up—only a small bound book. "What's this?" Smith asks.
    >
    > Oats, tidying up his clothing, pluckishly says, "That's my political study of voting habits in this district. Why don't you look at my lists? I work for important politicians now—guys with lots of clout." An ominous implication lurks in this last thrust.
    >
    > "Don't talk down to us," Smith snaps. But studying Oats's book, Jarvis finds nothing unusual. Smith finally hands him back his lists. Our cops can't hold him. Jarvis admits Oats can go. Just as a formality, Jarvis asks him, "Did you commit any criminal act in this building? Anything at all of which a courtroom jury could find you guilty?"
    >
    > "No," Oats says flatly. "No way," and jauntily skips off. Halting six blocks away, Oats digs a tiny picklock from his sock and a diamond ring from his shaggy hair.

2. Decrypt the message ALD CLO PXIB—BXQP XKVQEFKD XKA FP CLKA LC ZEFIAOBK which is in a generalized Caesar's Code.

3. Decrypt the message XDHF-BOOLBA BKEDQ-PYQFXEHBX XBAD knowing that the encoding funciton is $E(x) \equiv 7x + 1 \pmod{26}$.

4. Decrypt the following message which is a mono-alphabetic code:

   AIQTQVO QA BPM NIABMAB OZWEQVO XIZBQKQXIBQWV AXWZB WN UWLMZV BQUMA. QB QA VW TWVOMZ BPM MFKTCAQDM XZMAMZDM WN BPM DMZG ZQKP, VWZ LWMA WVM PIA BW JM EMITBPG BW WEV WVM'A WEV KZINB. AIQTQVO QA I NIUQTG AXWZB. BPMZM QA VW ZWWU NWZ BPM OMVMZIBQWV OIX QV I JWIB. EPMBPMZ GWCZ XZMNMZMVKM QA NWZ BPM NMMT WN I KIXZQKQWCA JZMMHM QV GWCZ KIVDIA, WZ NWZ BPM PMILG MFKQBMUMVB WN I ACZOQVO XWEMZ JWIB, EPMBPMZ GWC NQVL GWCZAMTN LZIEV QZZMAQABIJTG BW BPM WXMV AMI, WZ BISM GWCZ XTMIACZM QV QVTIVL EIBMZA, BPMZM QA VWBPQVO YCQBM TQSM BPM BPZQTT WN JMQVO QV AWTM KWUUIVL WN GWCZ WEV JWIB.

5. Determine which of the following have decoding functions, and for those that do, find the decoding function.

   (a) (For a 637 letter 'alphabet'). $E(x) \equiv 14x + 9 \pmod{637}$.

   (b) (For a 126 letter 'alphabet'). $E(x) \equiv 55x + 12 \pmod{126}$.

6. In Section 5.5 of *Ecco*, suppose we do not have the desirable property that $E_a(E_b(x)) = E_b(E_a(x))$. Amalgamated and Behemoth still want to send their signed contract to the lawyers. They are supposed to verify the copies before forwarding them. Explain how this can be done, if the lawyers are allowed to know the content of the contract?

7. In Section 6.3 of *Ecco*, suppose two of eight couriers may be double agents. What is the minimum number of parts into which the message must be divided in order that our agent can get the complete message while the enemy cannot?

8. In Section 6.3 of *Ecco*, suppose two of six couriers may be double agents. What is the minimum number of parts into which the message must be divided in order that our agent can get the complete message while the enemy cannot?

## 3.4 Questions

1. Decrypt the following message which uses the treasure hunt designer's code in Section 4.3 of *Ecco*: WFI DFIV GLQQCZEX RUMVEKLIVJ FW UI. VTTF, JVV KYV RLKYFI'J "TFUVJ, GLQQCVJ, REU TFEJGZIRTP", GLSCZJYVU SP N. Y. WIVVDRE.

2. Julius Caesar used the linear code with encoding function $E(x) = 9x + 5$. Decrypt his message: Z XFJP, Z LFV, Z XBSTDPCPG.

3. Solve the following problem which uses a mono-alphabetic code.

   C KBCDD MSJMQK RCK C MUJLCSF FQBOUJ GT RGJKUK CFP JSPUJK. OULWUUF LRUB LRUJU CJU 50 TUUL CFP 18 RUCPK. SF CPPSLSGF, LRU MSJMQK RCK KGBU AQFEDU CFSBCDK LRCL RCVU, CDLGEULRUJ, 11 RUCPK CFP 20 TUUL. LRUJU CJU LWSMU CK BCFY TGQJ–TGGLUP AQFEDU CFSBCDK CK LRUJU CJU LWG–TGGLUP AQFEDU CFSBCDK. RGW BCFY RGJKUK, JSPUJK, CFP AQFEDU CFSBCDK CJU SF LRU MSJMQK?

   YGQ KRGQDP RCVU DSLLDU PSTTSMQDLY PULUJBSFSFE LRCL LRUJU CJU 7 RGJKUK CFP 11 JSPUJK. OQL WRUF YGQ LJY LG KGDVU TGJ LRU FQBOUJ GT AQFEDU CFSBCDK, YGQ BCY OU KQJHJSKUP LG TSFP LRCL YGQ UFMGQFLUJ C FUECLSVU FQBOUJ. KGDVU LRU HJGODUB OY JSPPSFE YGQJKUDT GT CF QFWCJJCFLUP CKKQBHLSGF.

   Letter count:

   | U | C | J | L | F | K | G | S | R | Q | P |
   |---|---|---|---|---|---|---|---|---|---|---|
   | 65 | 46 | 38 | 38 | 37 | 34 | 30 | 28 | 25 | 24 | 22 |
   | D | B | T | Y | M | E | O | W | A | V | H |
   | 21 | 16 | 13 | 11 | 10 | 9 | 8 | 6 | 5 | 5 | 3 |

4. Decrypt the following passage which uses a mono-alphabetic code:

   E TSWA-EJB-LEFA MAHELSKJNRSP XSLR UKZM JASTRDKZM SN ANNAJLSEH LK LRA QZHH AJGKUIAJL KQ UKZM TEMBAJ. JAWAM CMSLSCSNA RSN CMAELSWA AQQKMLN, KM SJNSNL KJ TSWSJT RSI ZJNKHSCSLAB EBWSCA. XRAJ RA SN SJ LMKZDHA, DA PMAPEMAB LK RAHP KZL KQ CKZMNA, DZL UKZ NRKZHB MANPACL RSN PMSWECU EL EHH LSIAN. SQ RSN EJSIEH NRKZHB NLMEU, HAL RSI FJKX DU EHH IAEJN, DZL BKJ'L DA WSJBSCLSWA. EWKSB CKJNLEJL DKMMKXSJT. E RKZNARKHBAM SN AJLSLHAB LK MAIKWA EJU KDGACL LREL AJCMKECRAN KJ RSN PMKPAMLU, DZL SL IZNL DA MALZMJAB LK LRA MSTRLQZH KXJAM.

The frequency table is given below:

| A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 48 | 13 | 13 | 11 | 23 | 2 | 2 | 18 | 8 | 28 | 35 | 40 | 27 |
| N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| 27 | 0 | 8 | 7 | 21 | 39 | 7 | 9 | 0 | 10 | 5 | 0 | 17 |

5. Alice and Bob live many miles apart, and they wish to play a card game with an ordinary deck of 52 cards. To play the game, each must be "dealt" a hand with five cards, and they intend to accomplish this by mail. They will then play the game by telephone.

   They have one card in each of 52 identical boxes. Alice has 52 identical padlocks which can only be opened by her key, and Bob has 52 padlocks which can only be opened by his key.

   Each can put a padlock on as many boxes as they want. The boxes cannot be opened unless there are no padlocks on it. Alice has the 52 boxes at the start, but they can be mailed back and forth between the two of them.

   Design a protocol by which a hand consisting of 5 cards can be dealt to each player, such that

   (a) Neither player can determine the cards in the other player's hand.

   (b) During the play of the game, the players can announce what cards they hold themselves, and

   (c) at the end of the game the loser can verify (via mail) that the winner actually held the cards that he or she claimed.

6. In Section 6.3 of *Ecco*, suppose there are 7 couriers, and at most 2 may be captured. What is the minimum number of parts into which the code must be broken in order to get it safely to our agent?

7. A linear encoding function for a 34 letter alphabet is given by $E(x) \equiv 15x+7 \pmod{34}$. Find the decoding function.

8. Solve Problem #8 of the Omniheurist's Contest in Section 6.3 of *Ecco*.

9. Solve Problem #2 of the Omniheurist's Contest in Section 3.1 of *Ecco*.

# 4 Recursion & Induction

**Read Sections 1.2, 5.1, 5.2, 6.1 and 6.2 of *Ecco*.**

## 4.1 Recursion

In Railroad Blues (section 5.2 of *Ecco*), we were asked to turn a train around with as few uncouplings and couplings as possible. Here is a summary of the problem.

**Railroad Blues.** A single track with sidings at each end connects a steel mill to an iron ore mine. The train has one locomotive, 18 freight cars, and a caboose. The sidings (see the picture in *Ecco*) can only hold up to 13 items, whether the item is a freight car, a caboose, or the locomotive. What is the minimum number of uncouplings and couplings that are required to turn the train around?

If no uncouplings were permitted, we could only turn a train with 11 or fewer freight cars (the train has a locomotive and a caboose as well as freight cars). Dr. Ecco found that with 18 freight cars the task required two uncouplings and couplings. Actually, with that many uncouplings/couplings we can handle as many as 23 cars and a caboose. Moreover, if we eliminate the 30-car limit on the capacity of the track beyond the siding, Dr. Ecco's solution can be extended to allow us to turn trains of arbitrary length given a sufficient number of couplings/uncouplings. We summarize the results in the following table.

| Number of couplings/uncouplings | 0 | 2 | 4 | 6 | ... |
|---|---|---|---|---|---|
| Number of freight cars | 11 | 23 | 35 | 47 | ... |

The sequence $\{11, 23, 35, 47, \ldots\}$ may be defined as follows. The 0-th term is 11, and each subsequent term is obtained by adding 12 to the preceding one. Such a sequence is called an ***arithmetic progression***. The number 12 is called the ***common difference***.

Let us introduce some formal notation and denote the $n$-th term by $a_n$. Then

$$\begin{aligned} a_n &= a_{n-1} + 12, \text{ for } n \geq 1, \\ a_0 &= 11. \end{aligned}$$

The first equation is called a ***recurrence relation*** and the second is called the ***initial condition*** for the sequence $\{a_n\}$.

If we omit the initial condition, the recurrence relation alone defines a family of sequences, all having similar properties. In this case, they are all arithmetic progressions with common difference 12. By specifying an initial condition, we pick out one particular sequence from this family.

Let us get a clearer picture of what the recurrence relation means. The quantifier "for $n \geq 1$" means that the recurrence relation represents a sequence of equations:

$$\begin{aligned} a_1 &= a_0 + 12, \\ a_2 &= a_1 + 12, \\ a_3 &= a_2 + 12, \\ &\vdots \end{aligned}$$

Suppose we add the first three equations. Note that both $a_1$ and $a_2$ appear on both sides and can be cancelled. We are then left with $a_3 = a_0 + 12 \cdot 3 = 47$ as before. Another way of obtaining $a_3$ is by "repeated substitution" or ***iteration***:

$$a_3 = a_2 + 12 = (a_1 + 12) + 12 = (a_0 + 12) + 12 \cdot 2 = 47.$$

We did not make explicit use of the values of $a_1$ and $a_2$ in either of the two computations for $a_3$. While we could have computed the exact numerical value of $a_n$ for any $n$ from the recurrence relation with the initial condition, it may be desirable to able to do so from a general formula. This process is called ***solving*** the recurrence relation.

**Example 4.1.1.** *Solve the recurrence relation* $a_n = a_{n-1} + 12$ *with the initial condition* $a_0 = 11$.

*Solution.* Note that we have

$$\begin{array}{rlcccc} & a_1 & = & a_0 & + & 12, \\ & a_2 & = & a_1 & + & 12, \\ & a_3 & = & a_2 & + & 12, \\ & & \vdots & & & \\ & a_{n-1} & = & a_{n-2} & + & 12, \\ +) & a_n & = & a_{n-1} & + & 12, \\ \hline & a_n & = & a_0 & + & 12n. \end{array}$$

Using the initial condition, we have $a_n = 11 + 12n$. □

Alternatively, we can use iteration to solve the recurrence relation. We can begin the iteration either with $a_n$ and work our way back to the initial condition, or we can start with $a_1$ and work our way forward to $a_n$. It is sometimes just a matter of personal preference which iteration direction we use.

If we start with $a_n$ and work backwards to $a_0$ we get the following:

$$\begin{aligned}
a_n &= a_{n-1} + 12 \\
&= (a_{n-2} + 12) + 12 = a_{n-2} + 12 \cdot 2 \\
&= (a_{n-3} + 12) + 12 \cdot 2 = a_{n-3} + 12 \cdot 3 \\
&\vdots \\
&= (a_1 + 12) + 12 \cdot (n-2) = a_1 + 12 \cdot (n-1) \\
&= (a_0 + 12) + 12 \cdot (n-1) \\
&= a_0 + 12 \cdot n
\end{aligned}$$

If we start with $a_1$ and work our way to $a_n$, the iteration is:

$$\begin{aligned}
a_1 &= a_0 + 12 \\
a_2 &= a_1 + 12 = (a_0 + 12) + 12 = a_0 + 12 \cdot 2 \\
a_3 &= a_2 + 12 = (a_0 + 12 \cdot 2) + 12 = a_0 + 12 \cdot 3 \\
&\vdots \\
a_n &= a_0 + 12 \cdot n.
\end{aligned}$$

In both cases, we would complete the problem by using the initial condition and substitute 11 for $a_0$.

**Example 4.1.2.** *Find the sum of the first $n$ positive integers.*

*Solution.* We use an approach attributed to the great mathematician Gauss at age seven, although certainly the method has a much longer history. We denote the sum by $S_1$ and write it down twice, in different directions as shown below:

$$\begin{array}{rrcccccccccccc}
 & S_1 & = & 1 & + & 2 & + & 3 & + & \cdots & + & n, \\
+) & S_1 & = & n & + & (n-1) & + & (n-2) & + & \cdots & + & 1, \\
\hline
 & 2S_1 & = & (n+1) & + & (n+1) & + & (n+1) & + & \cdots & + & (n+1).
\end{array}$$

It follows that $S_1 = \frac{1}{2}n(n+1)$. □

The result in Example 4.1.2 can be restated as the following formula:

$$1 + 2 + 3 + \cdots + n = \frac{1}{2}n(n+1). \tag{4.1}$$

The expression on the left is called an ***open form*** while that on the right is a ***closed form***. The closed form is more convenient for computational use because it can be evaluated immediately by substituting the numerical value of $n$.

## The carrot cake

We now turn to the problem of the **carrot cake** in the Introduction to *Ecco*. Recall that the baker presented the young Jacob with the problem of cutting the cake into 16 equal pieces using four straight cuts.

Let $n$ denote the number of cuts, and $a_n$ the maximum number of pieces obtained. Clearly, the initial condition is

$$a_0 = 1,$$

and we should line up the pieces so that the next cut divides each into two. It follows that the recurrence is

$$a_n = 2a_{n-1}, \quad n \geq 1.$$

This process is called ***setting up*** the recurrence relation. We have to do this before we can solve a problem involving recursion.

**Example 4.1.3.** *Solve the recurrence relation* $a_n = 2a_{n-1}$ *with the initial condition* $a_0 = 1$.

*Solution.* By the method of iteration, we have

$$\begin{aligned} a_n &= 2a_{n-1} \\ &= 2(2a_{n-2}) \\ &= 2^2(2a_{n-3}) \\ &\vdots \\ &= 2^{n-1}(2a_0) \\ &= 2^n. \end{aligned}$$

□

The powers of 2 are an example of a ***geometric progression***, a sequence in which each term is obtained from the preceding one by multiplying it with a fixed number called the ***common ratio***.

**Example 4.1.4.** *Find the sum of the first* $n+1$ *powers of 2.*

*Solution.* Let $S$ be the sum $1+2+4+\cdots 2^n$. If we multiply $S$ by 2, each term of $S$ becomes the next one in $S$, that is, $1+2+4+\cdots+2^n$ is transformed into $2+4+8+\cdots+2^{n+1}$. If we now subtract $S$ from $2S$, we will get a lot of cancellations, as shown below:

$$\begin{array}{rrcrcrcrcrcrcrcl} & 2S & = & & & 2 & + & 2^2 & + & \cdots & + & 2^n & + & 2^{n+1}, \\ -) & S & = & 1 & + & 2 & + & 2^2 & + & \cdots & + & 2^n, & & \\ \hline & S & = & -1 & & & & & & & & & + & 2^{n+1}. \end{array}$$

It follows that $S = 2^{n+1} - 1$. □

The powers of 2 also feature prominently in the solution of the problem in Section 1.2 of *Ecco*. In that problem, millionaire Hank Alfred wanted to build a tower 1 km high from sections that are each 1 m high, and he asked Dr. Ecco how fast the task could be accomplished. If there are several stacks already constructed, it takes one week to place one stack on top of another — except when either or both of the stacks are more than 100 m high in which case it takes two weeks.

The idea was to build a lot of sections that are two metres high during the first week then use these to build sections that are four metres high during the second week and so on. Let $a_n$ denote the maximum height of a tower which can be built in $n$ weeks. Clearly, $a_0 = 1$ and $a_n = 2a_{n-1}$ for $1 \leq n \leq 7$. Since $a_7 = 128$, we do not have $a_8 = 2a_7 = 256$ because the stacks are now higher than 100. Instead, we have $a_9 = 256$.

How high a tower can we build in 8 weeks? If all of the previous sections that we have built are 128m, we cannot complete a tower during the eighth week. However, we would have no problem building stacks of height 100 in 7 weeks. It follows that we could have $a_8 = 200$. Following this scheme, we would have $a_n = 2a_{n-2}$ for $n \geq 10$. But then it would take 14 weeks to complete the task, and as Dr. Ecco has shown, by building some stacks smaller than the maximum height at each week, we can accomplish the task in 13 weeks.

## Carrot cakes again

The carrot cake problem becomes much more difficult if we impose the condition that no pieces are to be moved.

**The generalized carrot cake problem.** Into how many pieces can a cake be sliced using $n$ straight cuts if the pieces cannot be rearranged between cuts?

We can still obtain 1 piece with no cuts, 2 pieces with one cut, and and so on. The situation for the first few cuts is as follows:

| Number of cuts: | 0 | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| Number of pieces: | 1 | 2 | 4 | 8 | ? |

(The third cut is made horizontally.)

It looks like we should be able to get 16 pieces with 4 cuts, but it is not at all obvious how we can accomplish this. It is not easy to visualize how the cuts interact with one another, even when there are only 4 of them.

A useful strategy in problem solving is to consider a similar but simpler problem. Let us replace the carrot cake by a pizza, which may be treated as a two-dimensional object: Only vertical cuts are allowed.

**Example 4.1.5.** *What is the largest number of pieces you can get by cutting a pizza with $n$ straight vertical cuts if the pieces are not to be moved between cuts.*

*Solution.* We begin by making a table for 1, 2, 3, cuts and so on. It looks like this, where $n$ is the number of cuts and $a_n$ is the maximum number of pieces.

| $n$ | 0 | 1 | 2 | 3 | $\cdots$ |
|---|---|---|---|---|---|
| $a_n$ | 1 | 2 | 4 | 7 | $\cdots$ |

It is clear that $a_0 = 1$ and $a_1 = 2$. With two cuts, we can get either 3 or 4 pieces depending on whether the two cuts cross each other. Since we wish to maximize the number of pieces, we ensure that they cross. Hence $a_2 = 4$. Similarly, a third cut should cross both of the previous two, but not where they cross each other. This yields $a_3 = 7$. (Perhaps you might have expected $a_3$ to be eight! Already we have a glimpse of why we could not get 16 pieces in the carrot cake problem.)

Suppose $n-1$ cuts have been made and $a_{n-1}$ pieces are obtained. As before, the $n$-th cut should cross each of the others at distinct points. Hence there will be $n-1$ points of intersection on it, dividing it into $n$ segments. Each segment cuts an existing piece into two. Hence we get the recurrence relation

$$\begin{aligned} &a_n = a_{n-1} + n, \quad n \geq 1 \\ &a_0 = 1. \end{aligned}$$

We now solve this by iteration as follows, making use in the last step of the formula for $S_1$ in Example 4.1.2.

$$\begin{aligned} a_1 &= a_0 + 1, \\ a_2 &= a_1 + 2 = a_0 + 1 + 2 \\ a_3 &= a_2 + 3 = a_0 + 1 + 2 + 3 \\ &\vdots \\ a_n &= a_{n-1} + n = a_0 + 1 + 2 + \cdots + n \end{aligned}$$

So $a_n = 1 + \frac{1}{2}n(n+1)$.

This solves the pizza cutting problem. □

Before we can solve the generalized carrot cake problem, we need the following formula:

$$1^2 + 2^2 + \cdots + n^2 = \frac{1}{6}n(n+1)(2n+1). \tag{4.2}$$

The sequence of squares is neither an arithmetic progression nor a geometric progression. Hence the summation methods in Examples 4.1.2 and 4.1.4 do not apply. We have to find an alternative approach.

**Example 4.1.6.** *Find the sum of the squares of the first $n$ positive integers.*

*Solution.* Note that $(x+1)^3 = x^3 + 3x^2 + 3x + 1$. Letting $x = 1, 2, \ldots, n$ in turn, we have

$$\begin{array}{rrcccccccccc}
 & 2^3 & = & 1^3 & + & 3\cdot 1^2 & + & 3\cdot 1 & + & 1, \\
 & 3^3 & = & 2^3 & + & 3\cdot 2^2 & + & 3\cdot 2 & + & 1, \\
 & 4^3 & = & 3^3 & + & 3\cdot 3^2 & + & 3\cdot 3 & + & 1, \\
 & & \vdots & & & & & & & \\
 & n^3 & = & (n-1)^3 & + & 3(n-1)^2 & + & 3(n-1) & + & 1, \\
+) & (n+1)^3 & = & n^3 & + & 3n^2 & + & 3n & + & 1, \\
\hline
 & (n+1)^3 & = & 1 & + & 3S_2 & + & 3S_1 & + & n.
\end{array}$$

Here, $S_1 = n(n+1)/2$ from Example 4.1.2, and $S_2$ is the desired sum. From the last equation, we have

$$S_2 = \frac{1}{6}n(n+1)(2n+1). \qquad \square$$

***The solution of the generalized carrot cake problem***

Let $a_n$ denote the maximum number of pieces. Each straight cut is a plane, and if two planes intersect, they do so in a straight line. Suppose $n-1$ cuts have been made and $a_{n-1}$ pieces are obtained. The $n$-th cut should cross each of the other cuts along a distinct line. Hence there will be $n-1$ lines on this last plane. By Example 4.1.5, they divide the plane into $1 + \frac{1}{2}(n-1)n$ regions. Now each region divides an existing piece into two pieces, so

$$\begin{aligned}
a_n &= a_{n-1} + 1 + \frac{1}{2}n^2 - \frac{1}{2}n, \quad n \geq 1, \\
a_0 &= 1.
\end{aligned}$$

This recurrence relation may be solved as follows:

$$\begin{array}{rcccccccc}
 & a_1 & = & a_0 + & 1 + & \frac{1}{2}1^2 & - & \frac{1}{2}\cdot 1, \\
 & a_2 & = & a_1 + & 1 + & \frac{1}{2}2^2 & - & \frac{1}{2}\cdot 2, \\
 & & \vdots & & & & & \\
+) & a_n & = & a_{n-1} + & 1 + & \frac{1}{2}n^2 & - & \frac{1}{2}\cdot n, \\
\hline
 & a_n & = & a_0 + & n + & \frac{1}{2}S_2 & - & \frac{1}{2}S_1.
\end{array}$$

From the last equation, we have $a_n = \frac{1}{6}(n+1)(n^2 - n + 6)$.

Consider now the problem of the gold market in Section 5.1 of *Ecco*. It is reasonable to expect that whether the gold price goes up, stays put or goes down may depend what happens on what it did the day before.

**Example 4.1.7.** *If the price of gold does not change on a certain day, it will remain the same the next day with a probability of 0.1, go down with 0.4 and go up with 0.5. If it goes down on a certain day, it will continue to do so the next day with a probability of 0.6, stay put with 0.1 and go up with 0.3. If it goes up on a certain day, it will continue to do so the next day with a probability of 0.7, stay put with 0.1 and go down with 0.2. Let $a_n$ denote the probability that the gold price goes up on day $n$ and suppose the gold price goes up on day 0. Set up a recurrence relation for $\{a_n\}$ and solve it.*

*Solution.* The initial condition for this recurrence relation is $a_0 = 1$ because on day zero the price of gold went up. Then $a_1 = 0.7$. On the $(n-1)$-st day for $n \geq 1$, the gold price goes up with probability $a_{n-1}$, remains the same with probability 0.1 and goes down with probability $1 - a_{n-1} - 0.1 = 0.9 - a_{n-1}$. It follows that $a_n = 0.7a_{n-1} + 0.5 \cdot 0.1 + 0.3(0.9 - a_{n-1}) = 0.4a_{n-1} + 0.32$. That is, the recurrence relation is

$$\begin{aligned} &a_n = 0.4a_{n-1} + 0.32, \quad n \geq 1, \\ &a_0 = 1. \end{aligned}$$

For $n \geq 1$, iteration yields

$$\begin{aligned} a_n &= 0.4a_{n-1} + 0.32 \\ &= 0.4(0.4a_{n-2} + 0.32) + 0.32 \\ &= (0.4)^2(0.4a_{n-3} + 0.32) + 0.32(1 + (0.4)) \\ &= (0.4)^3(0.4a_{n-4} + 0.32) + 0.32(1 + (0.4) + (0.4)^2) \\ &= \cdots \\ &= (0.4)^{n-1}(0.4a_0 + 0.32) + 0.32(1 + (0.4) + (0.4)^2 + \cdots + (0.4)^{n-2} \\ &= \frac{0.32 + 0.1(0.4)^n}{0.6}. \quad \square \end{aligned}$$

## 4.2 Induction

The formula (4.2) on page 56 is quite complicated:

$$1^2 + 2^2 + \cdots + n^2 = \frac{1}{6}n(n+1)(2n+1).$$

it is understandable if we feel a little uneasy about the answer. We can do some checking by plugging in some numbers, and the results will be encouraging, but of course it does not really prove anything, and this is the purpose of this section. *We seek a way of justifying our solutions to recurrence relations.* There is a very basic and yet powerful method called ***mathematical induction***. We shall introduce this approach by examining Example 4.1.1.

By the method of iteration, we have obtained $a_n = 11 + 12n$ as the solution to the recurrence relation

$$\begin{aligned} a_n &= a_{n-1} + 12, \quad n \geq 1, \\ a_0 &= 11. \end{aligned}$$

What we want to do is to verify that the solution is valid for every $n$ greater than or equal to 0.

To help us achieve our objective, we shall build two voucher-issuing machines. The first machine, which we call the *base-case machine*, will issue a voucher with #0 on it if our solution is correct for the value $n = 0$.

The second machine, which we call the *induction machine* does the following: when we feed it a voucher, the induction machine verifies that the next one should be issued and does so. This way, we are guaranteed to eventually get voucher number $n$ for any $n$.

Let us see in practice what these machines are for Example 4.1.1.

The base-case machine simply checks that the solution is valid for the case $n = 0$. For $n = 0$, the formula $11 + 12n$ evaluates to 11, which agrees with the initial condition. So the base-case machine issues voucher number 0.

The induction machine is nothing but the given recurrence relation. We have already earned voucher number 0. Let us put it in. The machine accepts the value $a_0 = 11$ and computes $a_1 = a_0 + 12 = 11 + 12 = 23$. Since this agrees with the value given by our formula $(11 + 12 \cdot 1 = 23)$, the machine issues voucher number 1.

Suppose we have received voucher number 99. *This means that our formula gives the correct value for $a_{99}$, namely $a_{99} = 11 + 12 \cdot 99 = 1199$.* When we put voucher number 99 into the machine, it will accept our value of $a_{99}$ and compute $a_{100}$ by adding 12 to $a_{99}$:

$$a_{100} = a_{99} + 12 = 1211.$$

Our formula for $a_{100}$ says the value should be $11 + 12 \cdot 100 = 1211$, and since the two agree, the machine will issue voucher number 100.

Are we any further ahead than before? It does not seem so up to now. Let us say that we have got voucher $k$ for some nonnegative integer $k$. When we put this voucher into the machine, it accepts that $a_k = 11 + 12k$ and computes $a_{k+1}$ from the recursion formula:

$$a_{k+1} = a_k + 12 = 11 + 12k + 12 = 11 + 12(k + 1).$$

This agrees with our formula, and the machine issues voucher number $k + 1$.

Note that although we have not specified the value of $k$ we are assured that the induction machine will issue voucher number $k + 1$ when we feed voucher number $k$ into it. In other words, each voucher generates the next one, which is precisely what we want, and we now know that solution is valid for every integer $n \geq 0$.

To use this method for solving recurrence relations, we first must have a candidate for the solution. Of course, we can use iteration to find the candidate, but would we then need mathematical induction as well? You may recall that in the method of iteration, there is always a step in which we put in ellipses (the three dots "$\cdots$"). In a sense, this is not a formal way of proving things, and mathematical induction legitimizes that step.

There are many other problems which can be tackled by this approach. They invariably involve building the equivalent of two voucher-issuing machines.

The method of ***mathematical induction*** consists of two steps.

***Base case:*** We must prove that the first statement in the sequence is correct. This case is called the ***basis*** or ***base case***.

***Inductive step:*** We assume the validity of an arbitrary statement in the sequence and deduce that of the following one. The assumption is called the ***inductive hypothesis***, and the deduction, the ***induction argument***. It is good practice to state clearly the inductive hypothesis and also what you are trying to conclude.

**Example 4.2.1.** *Prove that* $1^3 + 2^3 + \cdots + n^3 = \frac{1}{4}n^2(n+1)^2$ *for all positive integers* $n$.

*Solution.* ***Base case:*** For $n = 1$, both sides are clearly equal to 1.

***Inductive step:*** First we state the inductive hypothesis: we assume that the statement is true for $n = k$. That is, assume that

$$1^3 + 2^3 + \cdots + k^3 = \frac{1}{4}k^2(k+1)^2.$$

Now we proceed with the induction argument: We want to show that the inductive hypothesis implies that the statement is also true for $n = k + 1$, that is, we want to show that

$$1^3 + 2^3 + \cdots + (k+1)^3 = \frac{1}{4}(k+1)^2(k+2)^2.$$

Indeed,

$$\begin{aligned} 1^3 + 2^3 + \cdots + (k+1)^3 &= (1^3 + 2^3 + \cdots + k^3) + (k+1)^3 \\ &= \frac{1}{4}k^2(k+1)^2 + (k+1)^3 \\ &= \frac{1}{4}(k+1)^2(k+2)^2. \end{aligned}$$

This completes the induction argument. □

A formal statement of the principle of induction is as follows:

**Theorem 4.2.2 (The principle of mathematical induction).** *Suppose that $X$ is the set of all integers that are greater than or equal to the integer $a$. Let $S$ be subset of $X$ with he following properties:*

(i) *$a$ is in $S$.*

(ii) *If $k$ is in $S$, then $k + 1$ is in $S$.*

*Then $S = X$.*

*Proof.* The proof of the principle of mathematical induction follows from the well ordering principle (see page 8). If $S$ does not equal $X$, then there are some integers which are in $X$ but not in $S$ and the well-ordering principle guarantees that among this set there is a smallest integer, say $m$, But $m$ cannot be $a$ because $a$ is in $S$. So $m > a$, and since $m$ is the smallest such integer, it follows that $m - 1$ must be in $S$. But then statement (ii) implies that $m$ is in $S$. This contradiction establishes the principle. □

Note that it is possible to strengthen the induction hypothesis. In trying to prove the result for the case $n = k + 1$, it is not necessary to assume only that case $n = k$ is valid. If we wish, we may assume that all of the cases $n = 0, 1, \ldots, k$ are valid. This is show-cased in the following example, which is related to the Knowledge Coordination problem in Section 6.2 of Ecco. The example also illustrates that *the application of mathematical induction is not restricted to proving formulae.*

**Example 4.2.3.** *An even number of girls are standing in a row, and a hat is put on each. None of them can see her own hat, but each can see the hats of all the girls with higher numbers than herself. They are told that all are wearing red hats, except for one who has a black hat on. Each is then asked in turn, starting with number 1, whether she can deduce the colour of her hat. They answer either "Yes" or "No". Prove that exactly one girl will say "No".*

*Solution.* Perhaps it does not seem that induction is useful here. What is the sequence of statements? The puzzle asks us to show that the conclusion is valid for every even number, that is, when there are two girls in a row, when there are four girls in a row, when there are six girls, and so on. Perhaps, then, the induction index is the number of girls in the row.

There is a second approach that we should consider. We may view the process as having a fixed number of girls, say $q$, where $q$ is even, and the puzzle is asking us to show that exactly one girl will say no if the black hat is on the first girl, if the black hat is on the second girl, if it is on the third girl, and so on. Here, the induction index is the position of the girl with the black hat.

Before beginning the induction process, note that at most one girl can say "No": As soon as one girl says "No" all of the girls following her can deduce that she does not see the black hat, and they will all conclude that they are wearing red hats.

Try the second approach. Let's see what happens if the black hat is on the girl in position $n = 1$.

Girl number 1 sees that all the other girls have red hats and can deduce that she has the black hat, so she says "Yes."

What about girl number 2? She has heard #1 answer "Yes", and she only sees red hats on the remaining girls. She realizes that whether she was wearing the black hat or not, girl #1 would answer "Yes". Girl #2 cannot deduce the colour of her hat, so she answers "No", and the remaining girls all answer "Yes".

What happens if the black hat is on girl #2?

Girl #1 sees the black hat, so she answers "Yes". Girl #2 is just as confounded as before—she does not know whether #1 has answered "Yes" because she has seen the black hat, or whether #1 has seen only red hats. So again girl #2 answers "No" and the remaining girls all answer "Yes".

Let us try one more case. What happens if the black hat is on girl #3?

Both #1 and #2 see the black hat, so they answer "Yes". Girl #3 sees only red hats, and she knows that if the black hat were on #1 or #2, then girl #2 would have said "No". Girl #3 therefore concludes that she is wearing the black hat and she answers "Yes". Girl #4 sees only red hats, and she realizes that whether she or girl #3 had the black hat, then girl #3 would say "Yes". So girl #4 answers "No", and the remaining girls all answer "Yes"

It looks like the following situation is happening: If the girl wearing the black hat is in an even position, then she will say "No". If she is in an odd position, then the girl after her will say "No".

We divide the girls into consecutive pairs and prove our claim by mathematical induction on $n$, where the black hat is worn by either girl in the $n$-th pair. In particular, we claim that if one of the girls in the $n$-th pair is wearing the black hat, then girl number $2n$ will be the first one to say "No."

***Basis:*** $n = 1$: We have already seen that the conclusion is true in this case.

***Inductive step:*** Assume that the conclusion is true for $n = 1, 2, \ldots, k$.

We want to show that if either of the girls in pair number $(k + 1)$ is wearing the black hat, then the first girl of the pair (girl number $2k+1$) will say "Yes" and the second (girl number $2k + 2$) will say "No".

Both girls of the pair can see that girls in position $2k + 3$ and beyond are wearing red hats. They can also hear that everyone before them has said "Yes" since girls 1 through $2k$ can see the black hat. By the induction hypothesis, both girls know that the $2k$ girls before them all wear red hats. Hence they can deduce that the black hat is on one of them. Since number $2k + 1$ can see the hat of number $2k + 2$ but not vice versa, the former will say "Yes" and the latter will say "No". This completes the inductive argument. $\square$

Although a problem may ask for the proof of a sequence of statements, it is not always necessary or advantageous to use mathematical induction. The conclusion of Problem 1 in the first Knowledge Coordination problem of *Ecco*(Section 6.1) may be rephrased as: "The generals will not attack after the $n$-th successful acknowledgement/counteracknowledgement." It is not clear how an inductive argument will go in this case. We have to come up with a general solution that works for all values of $n$, and make a machine that issues all the vouchers in one swoop!

## 4.3 Exercises

1. Solve the recurrence relation $a_n = 3a_{n-1} - 1$, with the initial condition $a_0 = 1$.
2. Find a closed form for $1 + 3 + 3^2 + 3^3 + \cdots + 3^n$.
3. Find a closed form for $1^3 + 2^3 + 3^3 + \cdots + n^3$.
4. Find a closed form for $\frac{1}{1} + \frac{2}{2} + \frac{3}{4} + \cdots + \frac{n+1}{2^n}$.
5. There are $2n$ participants in a chess tournament. In the first round, each plays exactly one game. Prove that the pairings for these $n$ games can be chosen in $1 \cdot 3 \cdot 5 \cdots (2n-1)$ ways.
6. An old puzzle called ***The Tower of Hanoi*** consists of three pegs, $A$, $B$, and $C$. On $A$ there are $n$ disks of different diameters arranged by decreasing size from the bottom to the top. You wish to transfer these from $A$ to $B$ using the smallest number of moves possible. The rules are as follows:

   Only one disk may be moved at a time, and it may be moved from one peg to either of the other two.

   No disk may be placed on one of a smaller diameter.

   (a) Explain recursively how to accomplish this. That is, assuming you can accomplish the task for $k$ disks, explain how to do it for $k + 1$ disks. Of course, include an initial case.

   (b) Let $a_n$ denote the smallest number of moves needed for $n$ disks. Write the recurrence relation for $n$, that is write down $a_1$, and express $a_{n+1}$ in terms of $a_n$.

   (c) Solve the recurrence in any way you can.
7. Observe that

$$\begin{aligned} 1 &= 1, \\ 1-4 &= -(1+2), \\ 1-4+9 &= 1+2+3, \\ 1-4+9-16 &= -(1+2+3+4). \end{aligned}$$

   Guess the general rule suggested by these examples, express it in suitable mathematical notation and prove it.

8. For which positive integers $n$ is it true that $2^n > 2n + 1$?

9. For which positive integers $n$ is it true that $2^n > n^2$?

10. Prove that the solution to the recurrence relation in Exercise 6 is $a_n = 2^n - 1$ by the method of mathematical induction.

11. **A card mystery.** You have a pack of cards numbered consecutively from 1 to n, with 1 being on the top of the deck. You deal as follows: Place the top card on the table, put the next card on the bottom of the pack, the next card on the table, the next card on the bottom of the pack, and so on. You keep doing this until you only have one card left. What is the number on that card? Show that your answer is true for all positive integers $n$ greater than 3.

12. What is wrong with the following inductive "proof" that all horses are the same colour.

    We will prove this by showing that every set of $n$ horses contains only horses of the same colour.

    *Base Case* ($n = 1$). It is clear that all horses in a set containing one horse are the same colour.

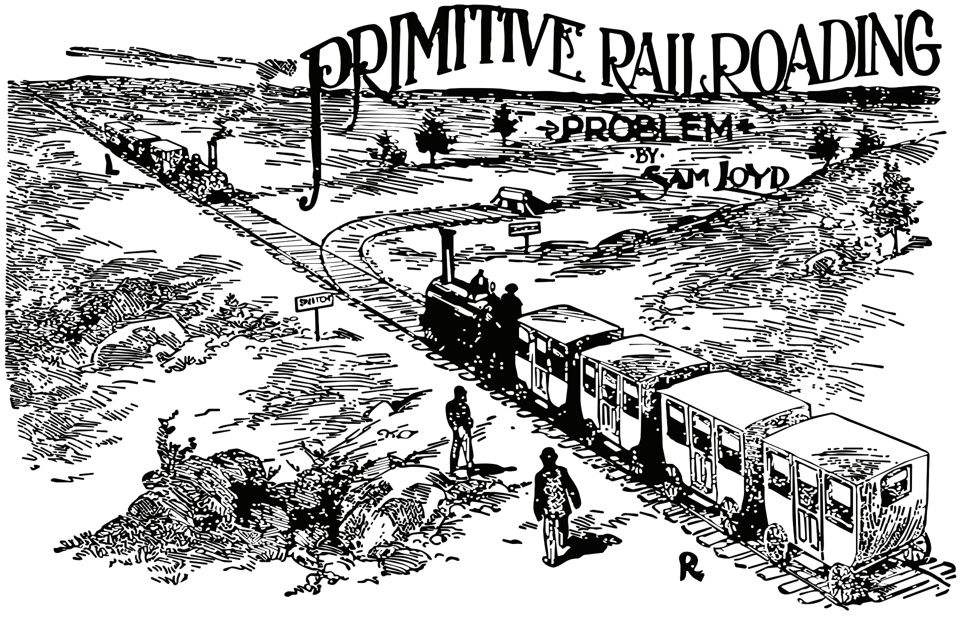
    *Inductive hypothesis*. Assume the theorem is true for $n = k$.

    *Inductive step*. Let $S$ be a set of $k + 1$ horses. Let the last two horses be labelled $h_k$ and $h_{k+1}$. Remove $h_{k+1}$ from $S$. This leaves a set $S'$ of $k$ horses, which are therefore all the same colour, say $C$. That is, horses $h_1, h_2, \ldots, h_k$ are all coloured $C$. It suffices to show that $h_{k+1}$ is also coloured $C$. Remove $h_k$ from the set $S'$ and replace $h_k$ with $h_{k+1}$ forming the set $S''$. By the inductive hypothesis, all of the horses in $S''$ must be the same colour. But this means that $h_{k+1}$ must be the same colour as the other horses in $S''$ and they are all coloured $C$. Therefore $h_{k+1}$ is coloured $C$, and the proof is finished.

## 4.4 Questions

1. For Problem 2 of Section 1.2 of *Ecco*, devise an algorithm to accomplish the task in 20 weeks.

2. **A variation of a Sam Loyd puzzle**



In this specimen of primitive railroading we have an engine and four cars meeting an engine with three cars. The problem is to ascertain the most expeditious way of passing the two trains by means of the side-track, which is only large enough to hold one engine or one car at a time. No ropes, poles or flying switches are to be used, and it is understood that a car *cannot* be connected to the front of an engine. How many couplings and uncouplings are required if each train has an engine and $n$ cars. (Each uncoupling must be accompanied by a coupling in order that the trains end up intact, so you need only count the couplings.) After passing, the order of the cars on each train must be the same. Write down a recurrence relation with initial conditions, explain why it works, and solve it.

Modification: How would you do it if there were $n$ and $n-1$ cars? (If you answer the modified question, you don't have to answer the original Sam Loyd Question.)

3. Ovals are closed curves in the plane that have the property that each two of them can intersect each other in at most two points. Determine the maximum number of regions into which the plane can be divided by $n$ ovals.

4. There are $n$ points on a circle, every two of which are joined by a chord. No three chords pass through a common point. Determine the number $a_n$ of points of intersections of these chords by means of a recurrence relation.

5. A certain basketball team can only sink foul shots and lay-ups, worth 1 and 2 points respectively. Let $a_n$ denote the number of ways the team can score $n$ points. (Scoring 1 then 2 is considered to be different than scoring 2 then 1). Write down a recurrence relation for $a_n$ with initial conditions for $a_0$ and $a_1$, and explain why it holds for all $n \geq 2$. **There is no need to solve the recurrence.**

6. Let $a_n$ be the number of 1's in the binary expansion of the positive integer $n$. Write down a recurrence relation for $a_n$ with initial conditions, and explain why it holds for all $n \geq 2$. **There is no need to solve the recurrence.**

7. Find a closed form for $1^4 + 2^4 + \cdots + n^4$.

8. Justify your result for question 7 by mathematical induction.

9. Find a closed form for $1 \cdot n + 2 \cdot (n-1) + 3 \cdot (n-2) + \cdots + (n-1) \cdot 2 + n \cdot 1$.

10. Justify your result for question 9 by mathematical induction.

11. Solve the recurrence relation $a_n = 2a_{n-1} + 1$, with the initial condition $a_0 = 1$.

12. A sequence $a_0, a_1, a_2, \ldots$ is defined by letting $a_0 = 3$ and
$$a_n = (a_{n-1})^2$$
for all $n \geq 1$. Show that $a_n = 3^{2^n}$ for all $n \geq 0$.

13. Show that
$$\left(1 - \frac{1}{2^2}\right)\left(1 - \frac{1}{3^2}\right)\left(1 - \frac{1}{4^2}\right)\cdots\left(1 - \frac{1}{n^2}\right) = \frac{n+1}{2n}$$
for all integers $n \geq 2$.

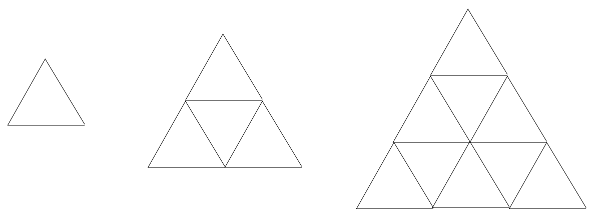
14. If the price of gold does not change on a certain day, it will remain the same the next day with a probability of 0.6, go down with 0.2 and go up with 0.2. If it goes down on a certain day, it will continue to do so the next day with a probability of 0.3, stay put with 0.6 and go up with 0.1. If it goes up on a certain day, it will continue to do so the next day with a probability of 0.3, stay put with 0.6 and go down with 0.1. During a certain week, the price goes up on Monday. What is the probability that it will go down on Friday?

15. On a certain planet, the weather each day is either good or bad. If it is good, then it remains good the next day 70% of the time. If it is bad, then it remains bad the next day 60% of the time. Assuming that as the number of days increases, the probability that the weather is good tends to a limit, calculate this limit.

16. In the Problem in Section 6.2 of *Ecco*, let the logicians, in alphabetical order, be numbered consecutively from 1 on. In a particular round, involving only those who have not decided, it is known that at least one of them is wearing an X. Suppose number $k$ is the last one who does. Prove that this logician is the first one who can decide in this round.

17. A game is played with a deck of $k$ cards numbered from 1 to $k$. They are shuffled thoroughly and the top card is turned over. If it is number 1, the game is won. If it is number $i$ where $2 \leq i \leq k$, then it is inserted into the deck so that it is the $i$-th card from the top. Then the new top card is turned over and the same process is applied. Can this game be won eventually, regardless of how the cards are stacked?

18. The figure below shows that 3 one–inch segments are needed to make an equilateral triangle of side 1, and 9 one–inch segments are needed to make an equilateral triangle of side 2 which is composed of 4 equilateral triangles of side 1, while 18 one–inch segments are needed to make an equilateral triangle of side 3 which is composed of 9 equilateral triangles of side 1. Let $a_n$ be the number of one–inch segments needed to



make an equilateral triangle of side $n$ which is composed of $n^2$ equilateral triangles of side 1. Find and solve a recurrence relation satisfied by $a_n$.

19. Using induction, show that you require at least $n+1$ integral weights to be able balance all integral weights from 1 to $2^n$.

20. (a) Find a closed form expression for $1 + 3 + 5 + \cdots + (2n - 1)$.

    (b) Make a conjecture about the terms of the following sequence, and prove your conjecture:
    $$\frac{1}{3}, \frac{1+3}{5+7}, \frac{1+3+5}{7+9+11}, \frac{1+3+5+7}{9+11+13+15}, \cdots .$$
    (*Note.* This sequence was used by Galileo in his work on freely falling bodies.)

# 5 Graph Theory

**Read Sections 1.3, 2.1, 2.3, 3.2, 3.3, 4.2, 4.4, 4.6, and 6.6 of *Ecco*.**

## 5.1 Basic properties

We start with the problem about the underground labyrinths in Section 1.3 of *Ecco*.

**The Odd Doors Problem.** Lawrence Terrence III has a problem. His recently departed father has hidden a cache of jewels in one of two underground labyrinths. All that Lawrence knows is that the jewels are in a room with an odd number of doors. His father told him that it is possible to determine from this fact which labyrinth contains the jewels. The only other information that Lawrence has is that the first labyrinth has two doors leading to the outside and the other has three.

Since it will cost a small fortune to explore each labyrinth, Lawrence Terrence III wants to know which labyrinth he should search.

We can draw some pictures that symbolize a labyrinth. We use dots to represent the rooms, and if two rooms are connected by a door, we draw a line between the dots to represent the door. If they are connected by two doors, we draw two lines between the two dots, and so on. Also, the world outside the labyrinth is represented by a dot since there are doors connecting each labyrinth to the outside world. One labyrinth will have three lines going to the outside dot, the other will have two lines. Here are some possible depictions of the labyrinths.

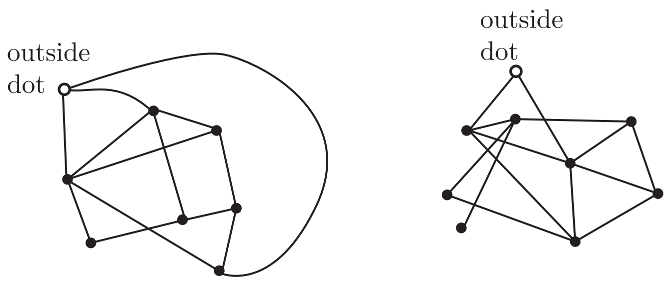


Figure 5.1. The Odd Doors Problem

A ***simple graph*** $G$ is formally defined as an ordered pair $(V, E)$ where $V$ is a nonempty set of elements called ***vertices*** and $E$ is a set of two-element subsets $e = \{u, v\}$ of $V$ called ***edges***. (The edge set $E$ can be empty.) Informally, a simple graph is just a collection of dots and lines, with at most one line joining a pair of dots, and with every line terminated by a dot at each end.

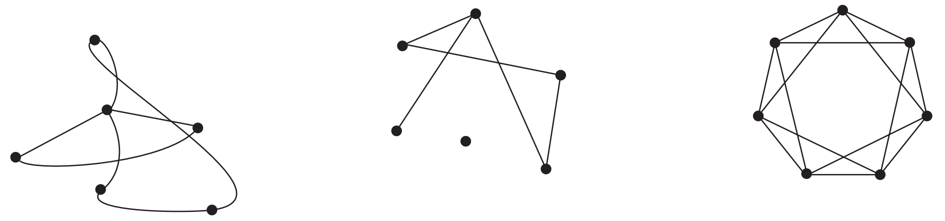


Figure 5.2. Simple graphs

There are occasions when we require something a bit more general. If some pairs of vertices have more than one edge joining them, the result is called a ***multigraph***, and if there are ***loops*** (which are edges beginning and ending at the same vertex), the result is called a ***pseudograph***. In Figure 5.3, (1) is a pseudograph, (2) is a multigraph, and (3) is a simple graph. In general, we will use the word "graph" to mean any of the three types.

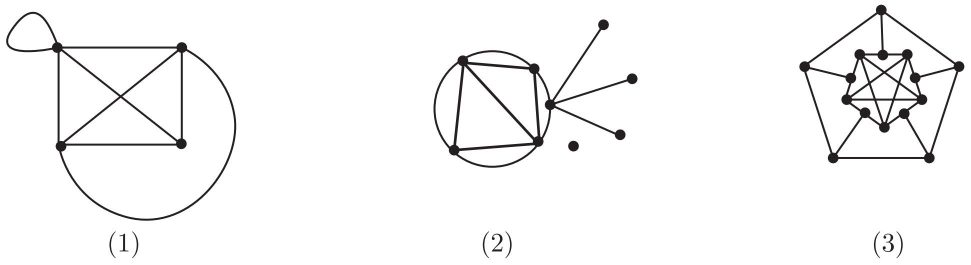


Figure 5.3. Various graphs

The graphs in Figure 5.1 depicting some of the possible labyrinths do not seem to help us a lot. The one that has three doors to the outside has five rooms with an odd number of doors, while the one with two doors to the outside has four rooms with an odd number of doors. Obviously, the father must expect his son to deduce that only one of the labyrinths is guaranteed to have a room with an odd number of doors, and that the jewels are in that labyrinth. But which one is it?

When we looked at the hypothetical graphs that represent the labyrinths, we counted the number of edges that are joined to each vertex. An edge that is joined to a vertex is said to be ***incident*** to the vertex, and the vertices that are joined to an edge are ***incident*** to the edge. More formally, the edge $e = \{u, v\}$ is ***incident*** with the vertices $u$ and $v$, which are in turn ***incident*** with $e$. We also say that $u$ and $v$ are ***adjacent*** to each other, and that they are the ***endpoints*** of the edge $\{u, v\}$. Two edges incident with the same vertex are also said to be ***adjacent***.

In a graph, the ***degree*** of a vertex $v$ is the number of edges incident to $v$, and is denoted $\deg(v)$. Each edge contributes 1 to the degree of each of the two vertices incident with it. We are now in position to state our first result.

**Theorem 5.1.1 (The Parity Theorem).** *The sum of the degrees of all vertices of a graph is equal to twice its number of edges.*

*Proof.* Each edge contributes 2 to the total degree. □

The Parity Theorem may be written as an equation: $2q = \sum_{v \in V} \deg v$, where $q$ is the number of edges. The Parity Theorem is also called the *Handshaking Lemma* and may be stated as follows:

**Theorem 5.1.2 (The Handshaking Lemma).** *Every graph has an even number of vertices of odd degree.*

Colloquially, this says that at a party the number of people who shake hands an odd number of times is even.

Note that both versions of the Parity Theorem are valid even if there are loops or multiple edges in the graph. It is clear from either version that the labyrinth with three outside doors is the one that is guaranteed to have at least one room with an odd number of doors.

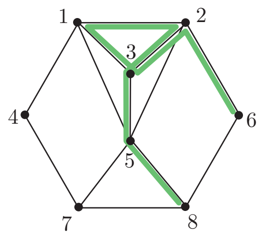
The simplicity of the definition of a graph makes it a most versatile tool. Graphs can be used as mathematical models in many scenarios. For the Spies and Acquaintances problem in Section 2.1 of *Ecco*, we can construct a graph where the vertices represent the spies, and the edges represent acquaintances. In the Party puzzle in Section 4.2, the vertices represent the people at the party and the edges represent handshakes.

### 5.1.1 Subgraphs

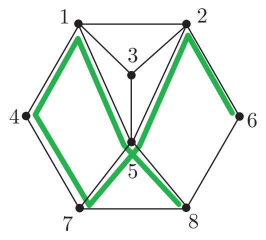
A ***subgraph*** of a graph is a subset of its vertices and edges, provided that all vertices incident with edges in the subgraph are included. In other words, a subgraph is a subset of the vertices and edges that itself forms a graph.

Certain types of subgraphs have specific names:

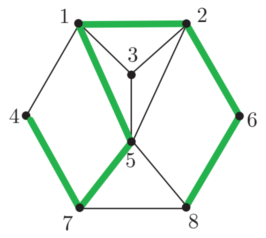
A ***walk*** is a subgraph that consists of a sequence of vertices and edges $v_0, e_1, v_1, e_2, v_2, \ldots, e_n, v_n$ such that for $1 \le i \le n$ the edge $e_i$ joins vertices $v_{i-1}$ and $v_i$. The walk on the right goes from vertex 8 to vertex 6 passing through the vertices 5, 3, 2, 1, 3, 2, 6. Note that it repeats vertices 3 and 2 and the edge $\{3, 2\}$.



A ***trail*** is a walk in which no edges are repeated. The trail depicted on the right starts at vertex 8 and passes through 5, 1, 4, 7, 5, 2, 6. Vertex 5 is repeated.



A ***path*** is a trail in which no vertices are repeated except perhaps for the first and last vertex. The path on the right joins vertex 4 to vertex 8 and passes through 7, 5, 1, 2, and 6.



A ***circuit*** is a trail whose first and last vertices are the same, and a ***cycle*** is a path whose first and last vertices are the same.

### Components of a graph

Two vertices of a graph that are joined by a path are said to belong to the same ***component*** of the graph. If the whole graph is one component, then it is said to be ***connected***. Thus the components are connected subgraphs which are not contained in larger connected subgraphs. We will be dealing with connected graphs most of the time. If a graph is not connected, we can usually consider one component at a time.

So far, we have not done much beyond introducing a host of definitions. While this mass of terminology may be daunting, it is fortunate that the language of graph theory is very intuitive and borrows extensively from everyday vocabulary.

## 5.2 Trees

A ***tree*** is defined to be a graph $T$ such that for any two vertices $u$ and $v$ in $T$, there is exactly one path which joins $u$ and $v$. Trees are one of the most important classes of graphs.

**Theorem 5.2.1.** *A tree has the following three properties.*

1. *It is connected.*
2. *It has no cycles.*
3. *It satisfies the* **Tree Formula** $V = E + 1$, *where* $V$ *and* $E$ *are the numbers of vertices and edges respectively.*

*Proof.* Since we can go from any vertex to any other vertex in a tree, it must be connected.

If it has a cycle, then for any two vertices on the cycle we can go from one to the other in at least two ways, which is a contradiction. Hence the first two properties are easily established.

The proof of the third property follows from the first two, and may be obtained by considering a physical model. Here, vertices are represented by beads and edges by threads. Pick one of the beads up and let the rest of the beads hang down. Since the tree is connected, all of the beads and threads are lifted up. Since the tree has no cycles, every bead other than the first is dangling at the end of exactly one thread, and of course every thread has exactly one bead dangling at its end. Hence the number of vertices is one more than the number of edges. □

---

**Kids and Chocolate bars**. What are the possible values of $n > 9$ such that $n$ children can equally share 9 identical chocolate bars, with the restriction that no bar be cut into more than two pieces.

---

Almost immediately you can see that $n = 18$ is possible—each of the eighteen children gets $\frac{1}{2}$ bar. You should also have little or no difficulty with the cases $n = 10$ and $n = 12$: for $n = 10$ each child receives a piece or pieces totalling $\frac{9}{10}$ of a chocolate bar, and for $n = 12$, each child receives the equivalent of $\frac{3}{4}$ of a bar.

So there are divisions for $n = 10, 12,$ and 18. It is clear that $n$ cannot be more than 18, for then each child would have to receive less than half a bar, and this would require that each bar be broken into more than two pieces.

If there are $n$ children, each child should get a piece or pieces that total $\frac{9}{n}$ of a bar, so it seems that we can do this by breaking each bar into $\left(\frac{9}{n}, \frac{n-9}{n}\right)$. The case $n = 11$ shows that we are not going to get off so easily. Each child must get pieces that total $\frac{9}{11}$ of a bar. Suppose we break each bar into $\left(\frac{9}{11}, \frac{2}{11}\right)$. Then nine children each get one of the $\frac{9}{11}$ pieces, but there is no way to combine the $\frac{2}{11}$ pieces to yield a total of $\frac{9}{11}$, for each of the remaining children.

This argument does not show that $n = 11$ is impossible because the it might be possible to accomplish the task with divisions other than $\left(\frac{9}{n}, \frac{n-9}{n}\right)$. Not all bars have to be divided the same way. For example, consider the case $n = 10$ again. Rather than dividing each bar into $\left(\frac{9}{10}, \frac{1}{10}\right)$, we could have divided them as $\left(\frac{9}{10}, \frac{1}{10}\right)$, $\left(\frac{8}{10}, \frac{2}{10}\right)$, etc., where the first child gets $\frac{9}{10}$, the second child gets $\frac{1}{10} + \frac{8}{10}$ and so on as follows:

$$(\underbrace{\tfrac{9}{10}}, \underbrace{\tfrac{1}{10}), \quad (\tfrac{8}{10}}, \underbrace{\tfrac{2}{10}), \quad (\tfrac{7}{10}}, \underbrace{\tfrac{3}{10}), \quad (\tfrac{6}{10}}, \underbrace{\tfrac{4}{10}), \quad (\tfrac{5}{10}}, \tfrac{5}{10}), \quad \cdots, \quad (\tfrac{2}{10}, \underbrace{\tfrac{8}{10}), \quad (\tfrac{1}{10}}, \underbrace{\tfrac{9}{10}})$$

Why not try the same strategy for the case $n = 11$? When we do, we run into a situation where a bar has to be divided into $\left(\frac{1}{11}, \frac{10}{11}\right)$, and one of the pieces is larger than the share each child should receive.

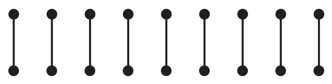
Perhaps this happens because for $n = 10$ there is *exactly one* child more than the number of bars, while for $n = 11$ there is *more than one* extra child. However, experimenting with $n = 12$ shows that this reasoning is also imperfect.

So now we can really understand the point of the problem:

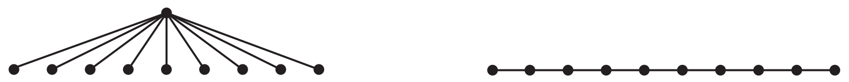
> *How come cases $n = 10, 12,$ and $18$ are easy to solve, while we can't seem to settle the case $n = 11$?*

Our task is now clear: we have to show that there is no solution for $n = 11$, that in fact there is only a solution if $n = 10, 12, \text{or } 18$.

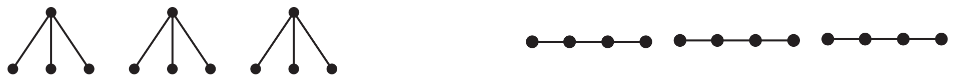
The problem can be solved by using a graph to represent the various cases, with the $n$ vertices being children and with an edge between two children if they share a chocolate bar. Note that each bar must be cut exactly once, so there are exactly 9 edges in the graph. The following are some of the graphs that correspond to solutions for $n = 18, 10,$ and $12$. There may be others, but note that they must all have exactly 9 edges.



$n = 18$



two graphs for $n = 10$



two graphs for $n = 12$

What are the properties of the graph that represents a solution? The first thing to notice is that the graph cannot contain a cycle. For example, if there were a 4-cycle in the graph, we have 4 chocolate bars (plus perhaps some other pieces) being distributed among 4 children, so the four children would get chocolate that amounts to a full bar (or more). This cannot be, because each child gets $\frac{9}{n}$ of a bar, where $n > 9$.

A graph that contains no cycles is either a single tree or a collection of disjoint trees (a ***forest***). If it is a single tree there are 9 edges and 10 vertices, and each child will get $\frac{9}{10}$ of a bar—and this was one of our solutions.

If it is a forest, and if one of the trees has $e$ edges (and therefore $e+1$ vertices), then each child associated with that tree gets $\frac{e}{e+1}$ of a bar. If another tree has $f$ edges, then the children associated with that tree would each get $\frac{f}{f+1}$ of a bar. Since all children receive the same amount of chocolate, it follows that $e = f$, that is, all of the trees in the forest must have $e$ edges. So if there are $k$ trees we must have $k \cdot e$ edges, and since there are nine bars, $k \cdot e = 9$. So the number of trees, $k$ must be a divisor of 9. This means that the forest contains either 1 tree, 3 trees, or 9 trees, corresponding to 10, 12 and 18 children. These are the only possible cases and so we have found them all.

### 5.2.1 When is a graph really a tree?

Recall that Theorem 5.2.1 showed that a tree is connected, has no cycles, and satisfies the tree formula. Neither of the three properties by itself defines a tree. A graph which is merely connected may have cycles, and is therefore not necessarily a tree. A graph without cycles is not necessarily connected, and may be the union of several trees, that is, it may be a forest. (Incidentally, a forest may consist of only one tree). The graph consisting of a triangle and a fourth edge disjoint from it has 5 vertices and 4 edges is not a tree but it does satisfy the Tree Formula.

On the other hand, in a connected graph there is at least one way to go from any given vertex to every other vertex. If the graph has no cycles, there is at most one way to go from any vertex to each other vertex. Hence the first two properties together are equivalent to our definition of a tree. Some books define a tree as a graph with these two properties.

Suppose a graph has the first and the third properties. If it has no cycles, then it is a tree. Suppose it has cycles. Discard an edge from a cycle. The resulting graph is still connected. Repeat this process until no more cycles remain. Then we have a tree, which satisfies $V = E + 1$. However, by assumption our graph satisfied this formula before any edges were removed. It follows that we have not removed any edges at all, and the original graph is already a tree.

In a similar way, you can show that a graph that has the second and third properties also must be a tree. So we can summarize:

**Theorem 5.2.2.** *A graph is a tree if it has any two of the following properties:*

1. *It is connected.*

2. *It has no cycles.*

3. *It satisfies the* **Tree Formula** $V = E + 1$*, where* $V$ *and* $E$ *are the numbers of vertices and edges respectively.*

For the Pebbles and Persuasion problem in Section 3.2 of *Ecco*, we can construct a graph where the vertices represent the charts and the edges represent support. It turns out that this graph is a tree, for good reasons: there cannot be cycles, as their existence implies circular reasoning. If the graph were not connected, this implies redundancy, which is also undesirable. Since everything leads towards the final conclusion in chart U, the vertex representing it may be referred to as the ***root*** of the tree, and tree itself a ***rooted*** tree.

In the Escaped Tiger of Section 4.6, we can construct a graph where the vertices represent the rooms and the edges represent the doors. Again we have a tree. If the graph is not connected, then parts of the temple need not be searched. If there are cycles, the keepers may end up chasing the tiger round and round.

The relevance of trees to the Code Invention problem of Section 4.4 is less apparent. We construct a rooted tree as follows. At the first level, we have just a single vertex which is the root, and it represents an empty word. Each vertex leads to two others in the next level, one at the other end of the edge going left, and one at the other end of the edge going right. If we proceed along an edge going left, the vertex we reach will represent a word obtained by appending a dot to the preceding word. If we proceed along an edge going right, we append a dash instead. Once a word has been chosen as a code word, no further edges emerge from the vertex representing it. This guarantees that the code generated is unambiguous.

## 5.3 Spanning Trees

Given a graph $G$, a ***spanning tree*** is a subgraph which is a tree and which contains all of the vertices of $G$.

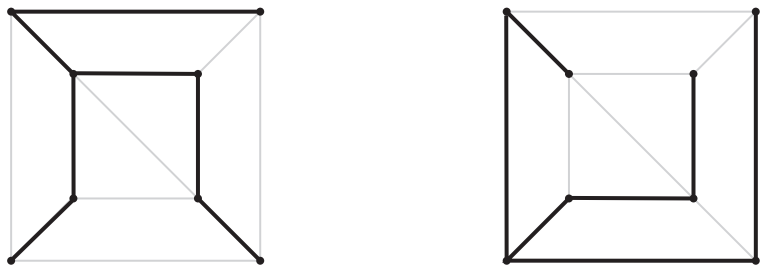


Figure 5.4. Two different spanning trees for the same graph.

If a graph has a spanning tree, the graph must be connected. It follows from our edge removal procedure in section 5.2.1 that the converse is also true.

**Theorem 5.3.1.** *A connected graph contains a spanning tree. In particular, in a connected graph, $E \geq V - 1$, where $E$ is the number of edges and $V$ is the number of vertices.*

**Example 5.3.2.** *A sheet of paper has been divided into many regions, each of which will be painted in one of 23 colours. Every colour will be used. Two colours are said to be neighbouring if they are used to paint two regions with a common boundary. Of course, there are many different ways to divide the paper into regions like this, but no matter how you do it, there will be neighbouring colours. What is the minimum number of neighbouring pairs of colours?*

*Solution.* Construct a graph with 23 vertices representing the colours. Join two vertices if and only if the corresponding colours are neighbouring. Clearly the graph is connected and the number of neighbouring colours is the number of edges in the graph. Letting $V$ and $E$ denote the number of vertices and edges, then $V = 23$, and $E \geq V - 1 = 22$, so there must be a least 22 neighbouring colours.

To show that this is really the minimum, we have to show that there is a way of dividing the paper into regions for which there are only 22 neighboring colours. Draw 22 mutually disjoint circles on the paper. The interiors of the 22 circles together with what's left over form 23 regions. Paint each of the 22 disks with a different colour and paint the left over region with the remaining colour. The number of neighbouring pairs of colours for this collection of regions is 22. □

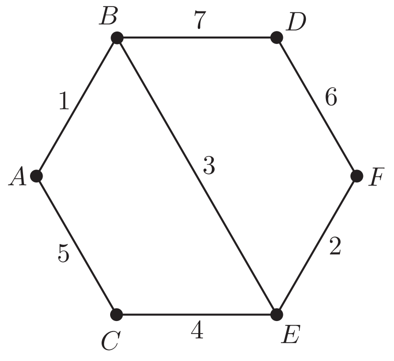
## 5.4 Kruskal's Algorithm

In some applications, the edges of a connected graph have a weight or length associated with them, and it is required to find a spanning tree whose total weight is as small as possible. Kruskal's Algorithm provides a systematic way of accomplishing this, and we present both a graphical and a tabular approach to the algorithm.

Suppose the graph $G$ has $p$ vertices and each edge $e$ in the edge set $E$ has a length denoted by $\ell(e)$. We wish to construct a spanning tree $T$ such that $\sum_{e \in E} \ell(e)$ is minimum. Note that there may be several such minimum spanning trees. For instance, if all the edges have length 1, then all spanning trees have total length $p - 1$.

In both approaches we begin by listing the edges in non-decreasing order of their lengths. We will use the following problem to illustrate both approaches

**Example 5.4.1.** *Find the minimal length spanning tree for the following graph.*



## Graphical approach

We begin by drawing a graph consisting only of the $p$ vertices of $G$. This graph has $p$ components (for Problem 5.4.1, $p = 6$).

We will add edges one by one. Each time we add an edge, it will join vertices in *different* components and so the number of components will diminish by 1. After we have added $p-1$ edges, we will have a tree.

We begin by listing the edges in increasing order of length (or, more accurately, in non-decreasing order):$AB$, $EF$, $BE$, $CE$, $AC$, $DF$, and $BD$.

Then we create a graph whose components are the vertices of the original graph. Figure 5.5 below shows the vertices forming the six initial components for Example 5.4.1. We now insert the edge of the shortest length. This is edge $AB$ and now we have a graph with five components (Figure 5.6).

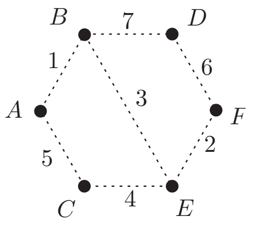


Figure 5.5.

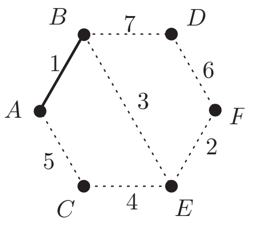


Figure 5.6.

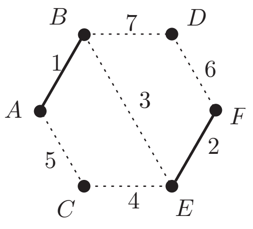


Figure 5.7.

The next step is to add the shortest edge that joins two of the 5 components—this is the edge $EF$ and we now have the graph shown in Figure 5.7 which has 4 components.

We continue in this fashion, continually adding the shortest edge between two of the existing components, creating the graphs shown in Figures 5.8, 5.9, and finally 5.10. Note that in Figure 5.9, the edge $AC$ is shorter than the edge $DF$, but we cannot use it because it joins two vertices in the same component.

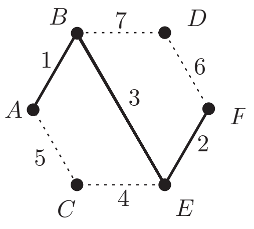


Figure 5.8.

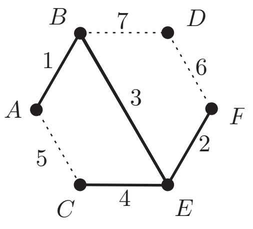


Figure 5.9.

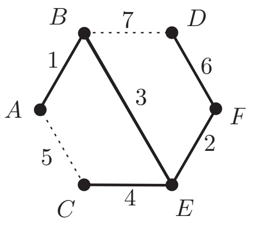


Figure 5.10.

***Justification of Kruskal's Algorithm***

It is clear from the construction that the final graph $T$ is a tree: at any stage during the algorithm, we never choose an edge which joins two vertices in the same component, so $T$ has no cycles. Since $G$ is connected, we can keep adding vertices until we have only one component. At that point, $T$ is also connected and it follows that $T$ is indeed a spanning tree of $G$.

We now prove that it has minimum total length. Label the edges $e_1$, $e_2$, …, $e_{p-1}$ in the order in which they are put into $T$. Note that

$$\ell(e_1) \leq \ell(e_2) \leq \cdots \leq \ell(e_{p-1}).$$

Suppose $T$ does not have minimum total length. Let $S$ be a minimum spanning tree of $G$ and suppose it contains $e_1$ through $e_k$ for some $k < p - 1$. If it does not even contain $e_1$, take $k$ to be 0. If there are several minimum spanning trees, choose $T'$ so that $k$ is as large as possible.

Let us add the edge $e_{k+1}$ to $S$. This forms a new connected graph, $S'$. Since $S$ has $p$ vertices and $p$ edges, it cannot be a tree and it must contain a cycle. The cycle contains an edge $e'$ which cannot be in $T$. In particular, $e'$ cannot be any of $e_1$ through $e_{k+1}$.

Now remove $e'$ from $S'$ forming a subgraph, $S''$, which is connected and has $p - 1$ edges. Hence $S''$ is also is a spanning tree of $G$. Note that $e'$ does not complete a cycle with any subset of $\{e_1, e_2, \ldots, e_{k+1}\}$ since all $k+1$ edges belong to $S'$. Note that we have transformed $S$ to $S''$ by deleting edge $e'$ and inserting $e_{k+1}$. All the other edges of $S$ and $S''$ are the same.

Now consider the stage in Kruskal's Algorithm where we have added edges $e_1$ through $e_k$. In the next step both $e_{k+1}$ and $e'$ were among the candidates for insertion into the graph and we chose $e_{k+1}$ over $e'$. This means that $\ell(e_{k+1}) \leq \ell(e')$ and so the total length of $S''$ is no greater than the total length of $S$. So either $S$ was not a minimal spanning tree or else $S$ was a minimal spanning tree but $k$ was not as large as possible. In either case we have a contradiction, and this justifies Kruskal's Algorithm.

## The tabular form of Kruskal's Algorithm

We illustrate a tabular approach which is more suited to computer use. As before, begin by listing all of the edges in non-decreasing order.

$$AB, \quad EF, \quad BE, \quad CE, \quad AC, \quad DF, \text{ and } BD.$$

Label all of the vertices from 1 to $p$, where $p$ is the number of vertices (for Example 5.4.1, $p = 6$). The labels denote the components that the vertices belong to, and at the initial stage, each vertex is a separate component.

We consider each edge in order. Suppose it joins two vertices with labels $p < q$. We put this edge into $T$ and change *all* labels $q$ into $p$. Suppose the edge joins two vertices with the same label. We pass over it and do not change any labels. We continue until all labels have become 1.

For Example 5.4.1 the following table summarizes the process. The vertices are the top row, and the successive rows indicate the component that the vertices belong to. The left most column contains the edges in increasing order of length.

| Vertices$\rightarrow$ | $A$ | $B$ | $C$ | $D$ | $E$ | $F$ | |
|---|---|---|---|---|---|---|---|
| Edges $\downarrow$ | 1 | 2 | 3 | 4 | 5 | 6 | (initial components) |
| $AB$ | 1 | 1 | 3 | 4 | 5 | 6 | (Insert $AB$ and change the label for $B$.) |
| $EF$ | 1 | 1 | 3 | 4 | 5 | 5 | (Insert $EF$ and change the label for $F$.) |
| $BE$ | 1 | 1 | 3 | 4 | 1 | 1 | (Insert $BE$ and change the labels for $E$ and $F$.) |
| $CE$ | 1 | 1 | 1 | 4 | 1 | 1 | (Insert $CE$ and change the label for $C$.) |
| $AC$ | | | | | | | (Skip $AC$: $A$ and $C$ have the same label.) |
| $DF$ | 1 | 1 | 1 | 1 | 1 | 1 | (Insert $DF$ and change the label for $D$.) |

At this point, all vertices have the same label and so we're finished. The minimum spanning tree $T$ consists of the edges $AB$, $EF$, $BE$, $CE$, and $DF$.

## 5.5 Dijkstra's Algorithm

Here is another problem, known as the *single source shortest path problem.* Let $G$ be a connected graph with $p$ vertices, and let $v_0$ be one of its vertices. We seek a spanning tree $T$ such that for any other vertex $v$, the shortest path $P$ in $G$ joining $v$ and $v_0$ is in $T$. We may regard $T$ as a tree with ***root*** $v_0$. $T$ is called a shortest-path spanning tree with root $v_0$. The solution given here is known as Dijkstra's Algorithm.

### The graphical version of Dijkstra's Algorithm

Start with the root. This subgraph is a tree $T_0$ in $G$. Then the algorithm identifies all vertices that are adjacent to the root (these are called the ***fringe*** vertices for $T_0$. It identifies the shortest edge (say $e_1$) that joins the root to a fringe vertex (say $v_1$) and adds $e_1$ and $v_1$ to $T_0$ forming a new tree, $T_1$.

Each step of the algorithm creates a new tree by adding a new edge and vertex.

Suppose that the tree $T$ has been created, where $T$ does not contain all the vertices of $G$. Identify the ***fringe vertices*** for $T$: these are the vertices in $G$ but not in $T$ that are adjacent to at least one vertex of $T$.

A fringe vertex, $w$, is depicted in Figure 5.11, where the heavy lines and solid dots form the tree $T$ and the open dot is a fringe vertex. The dashed lines are all of the edges that join the fringe vertex to vertices of $T$. The number next to a vertex is its distance from the root and the number next to an edge is the length of that edge. For each fringe vertex $w$, choose the edge that makes the path from $v_0$ to $w$ as short as possible. This is the associated ***candidate edge***. In Figure 5.11 this would be the edge of length 3. Note that this is not the shortest edge adjacent to $w$!.

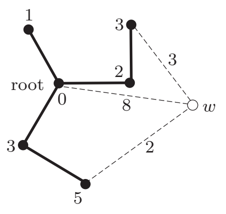


Figure 5.11.

Identify the candidate edge for every fringe vertex, and then choose the fringe vertex (and associated candidate edge) whose distance from $v_0$ is the smallest. Add these to $T$ to create a new tree.

Continue in this fashion until all vertices of $G$ have been included. The resulting tree is a shortest-path spanning tree with root $v_0$. There may be more than one such tree, but for each vertex $v$, the path lengths from $v_0$ to $v$ will be the same in all cases.

**Example 5.5.1.** *For the graph in Example 5.4.1, construct a spanning tree yielding shortest paths from the root $A$.*

*Solution.* The sequence of figures 5.12 through 5.16 shows how the tree is built up from the root.

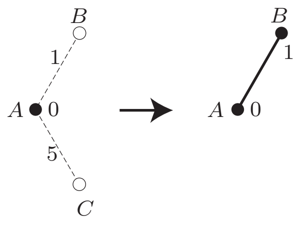


Figure 5.12.

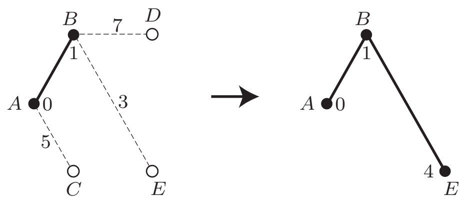


Figure 5.13.

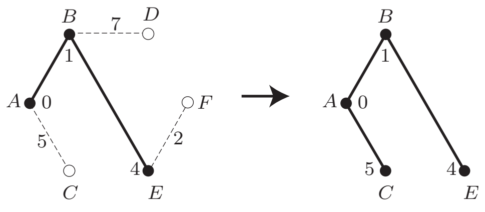


Figure 5.14.

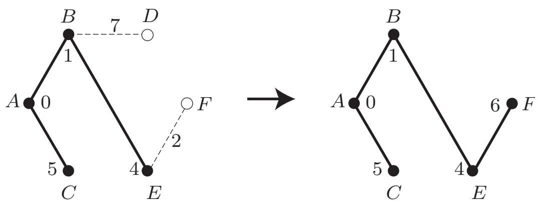


Figure 5.15.

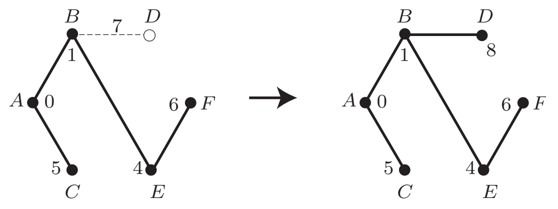


Figure 5.16.

The shortest-distance spanning tree $T$ rooted at $A$ consists of the edges $AB$, $BE$, $EF$, $AC$ and $BD$. It is different than the tree constructed by Kruskal's algorithm in Example 5.4.1. □

***Justification of Dijkstra's Algorithm***

Suppose $G$ has $p$ vertices. Let $T$ be the spanning tree of $G$ constructed by Dijkstra's Algorithm (it is clear that $T$ is a tree). Suppose that the vertices are added to $T$ in the order $v_0$, $v_1$, …, $v_{p-1}$ . For $1 \leq i \leq p-1$, we use the following notation to help us with our bookkeeping:

| | |
|---|---|
| $P_i$: | the path in $T$ joining $v_0$ to $v_i$. |
| $d_i$: | the length of $P_i$. |
| $\ell\{u, v\}$: | the length of the edge joining vertices $u$ and $v$. |

We claim that $P_i$ is the shortest among all paths in $G$ that connect $v_0$ and $v_i$. We use induction on $i$. The case $i = 1$ follows directly from the first step of the algorithm. Suppose the claim holds for $i = 1, 2, \ldots, j$. Consider the next case $i = j + 1$. Assume that there is a path $P$ in $G$ joining $v_0$ and $v_{j+1}$ which is shorter than $P_{j+1}$.

The path $P$ contains some of the vertices of the set $S = \{v_0, v_1, \ldots, v_j\}$. As we traverse $P$ from $v_0$ to $v_{j+1}$ let $v_k$ be the last vertex on $P$ which belongs to $S$, and let $v$ be the vertex on $P$ after $v_k$. The vertex $v$ is not on the path $P_{j+1}$ because the vertices of $P_{j+1}$ all belong to $S$. Let $v_t$ be the vertex immediately preceding $v_{j+1}$ on the path $P_{j+1}$. Since we chose $\{v_t, v_{j+1}\}$ over $\{v_k, v\}$ in Dijkstra's Algorithm, we must have

$$d_{j+1} = d_t + \ell\{v_t, v_{j+1}\} \leq d_k + \ell\{v_k, v\}.$$

By the induction hypothesis, $P_k$ is the shortest path in $G$ joining $v_0$ to $v_k$. Hence the length of $P$ is at least $d_k + \ell\{v_k, v\}$. That is, $P$ is not shorter than $P_{j+1}$. This contradiction justifies Dijkstra's Algorithm.

## The tabular form of Dijkstra's Algorithm

We begin by listing the vertices in the top row of the table and as the edges are added they will appear in the rightmost column of the table. An entry in column for vertex $u$ is of the form "$v, d$", where $u$ is a fringe vertex for $v$ and where $d$ denotes the length of the path in the tree from the root to the fringe vertex via $v$. In the first row of the table, only the root belongs to the tree, and the distance from every other vertex to the root is in effect infinite. So in row 0 of the table, the entry below $v_0$ is $-, 0$ and the entry below all other vertices is $-, \infty$.

Suppose that at some point we have inserted $v_0, v_1, v_2, \ldots, v_i$ and the associated edges into the tree, where $v_i$ is the last vertex appended and that, in the tree, the path length from $v_0$ to $v_i$ is $d_i$. The label in row $(i)$ below $v_i$ will be $u, d_i$ where $u$ is one of $v_0$ through $v_{i-1}$.

To construct row $(i + 1)$: If $v$ is already in the tree, the entry below $v$ is blank. If $v$ is not in the tree and is not adjacent to $v_i$, the label below $v$ is the same as it is in row $i$. If $v$ is not in the tree and is adjacent to $v_i$, suppose that the label below $v$ is $w, d$ (which could be $-, \infty$). Let $d' = d_i + \ell\{v_i, v\}$. If $d' < d$, in row $(i + 1)$ the label below $v$ becomes $v_i, d'$ otherwise it remains as it is in row $i$.

To find the vertex $v_{i+1}$ that is next inserted into the tree, find the entry $w, d$ in row $i + 1$ with the smallest $d$. The vertex that is at the head of this column becomes $v_{i+1}$ and the associated edge is $\{w, v_{i+1}\}$. The process stops after we have inserted all vertices.

The following table illustrates the process for Example 5.5.1. The rightmost column records the edges as they are inserted, and the rows are numbered from (0) to (5) to correspond to the description of the tabular algorithm.

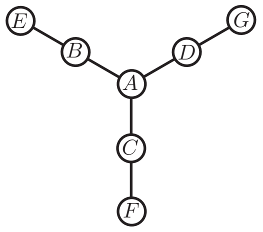
| $A$ | $B$ | $C$ | $D$ | $E$ | $F$ | $\leftarrow$ Vertices | |
|---|---|---|---|---|---|---|---|
| $-, 0$ | $-, \infty$ | $-, \infty$ | $-, \infty$ | $-, \infty$ | $-, \infty$ | $\downarrow$ Edges | (0) |
| | $A, 1$ | $A, 5$ | $-, \infty$ | $-, \infty$ | $-, \infty$ | $AB$ | (1) |
| | | $A, 6$ | $B, 8$ | $B, 4$ | $-, \infty$ | $BE$ | (2) |
| | | $A, 6$ | $B, 8$ | | $E, 6$ | $AC$ | (3) |
| | | | $B, 8$ | | $E, 6$ | $EF$ | (4) |
| | | | $B, 8$ | | | $BD$ | (5) |

### ***Greedy algorithms***

Both Kruskal's and Dijkstra's Algorithms are examples of a whole class known as ***Greedy Algorithms***. What such an algorithm does is to make the best possible choice at each stage. This leads to the best possible choice overall most of the time. However, it is not guaranteed to do so, even though what it produces is usually not too far off the optimal result either.

Here is an example where the greedy algorithm fails: Suppose seven villages are linked as follows. $A$ is at the centre and is linked to $B$, $C$ and $D$, which are in turn linked to $E$, $F$ and $G$. There are no other links. Now hospitals are to be built in these villages, such that each village either has its own hospital or is linked to a village which has one. We would like to build as few hospitals as possible.

Note that if we build one in $A$, it will serve $A$, $B$, $C$ and $D$. If we build one in $B$, $C$ or $D$, it will serve three villages. If we build one in $E$, $F$ or $G$, it will only serve two. Hence the Greedy Algorithm dictates that the first one built should be at $A$. However, we would then need three more hospitals, whereas one in each of $B$, $C$ and $D$ would have served the purpose.



## 5.6 Planar Graphs

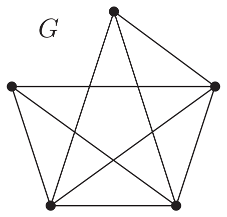
A graph is said to be ***planar*** if it can be drawn in the plane so that its edges only meet at the vertices. Unlike most properties of graphs which are purely combinatorial, planarity is a topological property. When drawn without crossing edges, a planar graph divides the plane into regions called ***faces***, including an infinite one.

It is possible to draw the same planar graph in many different ways without crossing edges, but its number of faces is always the same. A more specific form of this result is expressed in the following, the most basic property of planar graphs.

**Theorem 5.6.1 (Euler's Formula).** *In a connected planar graph, $V - E + F = 2$, where $V$, $E$ and $F$ are its numbers of vertices, edges and faces respectively.*

*Proof.* It is not difficult to prove this. First note that a tree is a connected planar graph. Since it has no cycles, it does not enclose any finite faces. Hence $F = 1$. By the Tree Formula, $V - E = 1$ so in this case $V - E + F = 2$. For a general planar graph, consider first a spanning tree for it. Euler's Formula holds at this point. Now add the remaining edges one at a time. Each time $E$ goes up by 1, $F$ also goes up by 1 since the new edge divides an existing region into two. When the planar graph is reconstructed, we have $V - E + F = 2$. □

It is not always obvious when a graph is a planar graph. For example, the graph $G$ in Figure 5.17 is planar, but it may take several tries before you can redraw it without crossing edges. The graph $H$ cannot be drawn without edges crossing, and if you try you will repeatedly fail. But repeated failures is not a proof—how can you be sure that you haven't overlooked some possibilities?



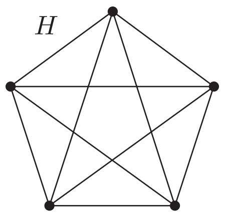


Figure 5.17.

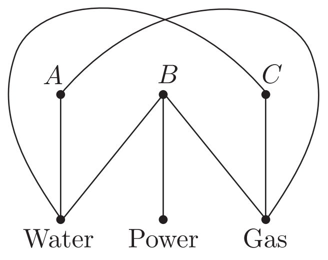
The graph $H$ is a special type of graph called a ***complete graph*** which is a graph in which every two vertices are joined by an edge. A complete graph with $n$ vertices is denoted $K_n$, so $H$ is usually called $K_5$. The following example uses the fact that every edge in a planar graph is incident with at most two faces. Although this fact seems intuitively clear when the graph is drawn in the plane without crossing edges, the proof is very difficult and is beyond the scope of these notes.

**Example 5.6.2.** *Prove that the complete graph $K_5$ on 5 vertices is non-planar.*

*Solution.* We have $V = 5$ and $E = 10$. If it is planar, then according to Euler's Formula, $F = 7$. Now each of the 7 faces has at least 3 edges on its boundary. So the total count of 'boundary edges' is at least $7 \cdot 3 = 21$. However, each of the 10 edges serves as a boundary edge for at most 2 faces, so that the total count is at most 20. We have a contradiction. It follows that $K_5$ cannot be planar. □

A ***bipartite*** graph is one whose vertices can be partitioned into two subsets such that no edge joins two vertices in the same subset. If each vertex in one subset is joined to every vertex in the other subset, the result is called a ***complete bipartite*** graph. $K_{n,m}$ denotes the complete bipartite graph with $n$ vertices on one side and $m$ on the other. The most infamous bipartite graph is the one in the well-known utilities problem:

**The utilities problem.** There are three side-by-side houses, $A$, $B$, and $C$, and three utility stations to supply them with water, power, and gas. The problem is that no matter how the city tries, the lines joining the utilities to the houses cross somewhere, like they do in the figure below. Is it possible to avoid this dilemma? (No lines are allowed to pass beneath a house or a supply station.)



The problem is asking whether $K_{3,3}$ is planar, which we now show is not the case.

**Example 5.6.3.** *Prove that the complete bipartite graph $K_{3,3}$ is non-planar.*

*Solution.* We have $V = 6$ and $E = 9$. If it is planar, then $F = 5$ by Euler's Formula. Note that every cycle in a bipartite graph must have an even number of edges, since the vertices on the cycle must come alternately from the two subsets. Hence each face has at least 4 edges on its boundary. So the total count of boundary edges is at least $5 \cdot 4 = 20$. On the other hand, it cannot be more than 18 since there are 9 edges. Again we have a contradiction. It follows that $K_{3,3}$ is also non-planar. □

If a graph is planar, clearly any of its subgraphs is also planar. Equivalently, if one of the subgraphs is non-planar, then the graph itself is also non-planar. Since $K_5$ and $K_{3,3}$ are non-planar, any graph containing either of them will also be non-planar.

What can we say about a graph which does not contain either $K_5$ or $K_{3,3}$ as a subgraph? It is tempting to conclude that it must be planar, and that is not too far from the truth. Before we can state an important result which says essentially that, we need a definition. A ***subdivision*** of a graph is a new graph obtained from the old one by subdividing some of its edges by inserting vertices of degree 2. In other words, we obtain a subdivision by placing new vertices on existing edges as in Figure 5.18.

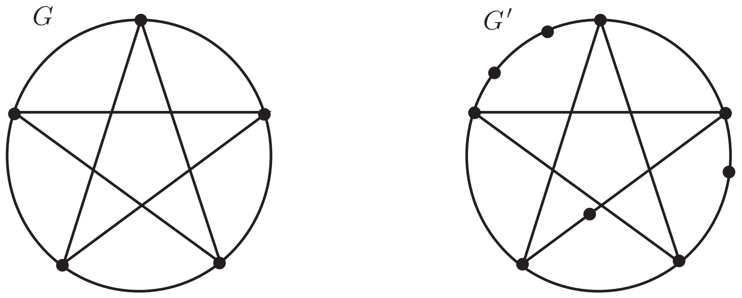


Figure 5.18. $G'$ is a subdivision of $G$

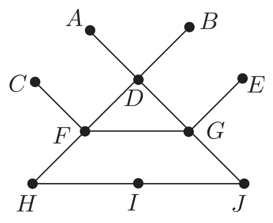
**Theorem 5.6.4 (Kuratowski's Theorem).** *A graph is planar if and only if it does not contain a subdivision of either $K_5$ or $K_{3,3}$.*

The proof of the "only if" part is easy. If a graph contains a subdivision of either of the two forbidden graphs, it is clearly not planar. The proof of the "if" part is very difficult. We omit it because for graphs that are not too complicated, the easiest way to show that it is planar is to draw it without crossing edges.

## 5.7 Centre, radius, and diameter

Let $G$ be an arbitrary graph. The ***distance*** between two of its vertices $u$ and $v$ is defined to be the number of edges on the shortest path joining $u$ and $v$. Consider the distance from $v$ to any other vertex $u$ of $G$. The maximum of these distances is called the ***radius*** of $v$. Consider the radius of any vertex $v$ of $G$. The minimum of these radii is called the ***radius*** of $G$, and a vertex $v$ is called a ***centre*** of $G$ if its radius is equal to the radius of $G$. Finally, consider the distance between any two vertices $u$ and $v$. The maximum of these distances is called the ***diameter*** of $G$.

**Example 5.7.1.** *For the graph in the diagram below, find the radius of each vertex as well as the radius, diameter and centres of the whole graph.*



*Solution.* The radii of $A$, $B$, $C$, $D$, $F$, $G$, $H$, $I$ and $J$ are 4, 4, 3, 3, 3, 2, 2, 3, 4 and 3 respectively. Hence the radius of the whole graph is 2, its diameter is 4 and its centres are $F$ and $G$. □

## 5.8 Exercises

1. There are 20 teams in a camp. Each day, they square off in 10 pairs. Prove that after the second day, we can choose 10 teams no two of which have yet played each other.

2. A country has more than 101 cities. The capital city is connected by direct flights to exactly 100 other cities. Each city other than the capital is connected by direct flights to exactly 10 other cities. At present, it is possible to travel by air from any city to any other, changing planes if necessary. Prove that this is still possible after shutting down 50 suitably chosen direct flights involving the capital city.

3. A host is expecting either 7 or 11 children. She has 77 marbles as gifts, which may be shared equally among the children. She puts them into bags, not necessarily containing the same number of marbles, so that whether 7 or 11 children turn up, each will get a number of bags containing a fair share. What is the minimum number of bags?

4. You are given pile of $n > 1$ pennies.

   I. Divide the pile into two piles, one containing $m$ and the other containing $k$ pennies where $m$ and $k$ are both positive.

   II. Write down the product $m \cdot k$.

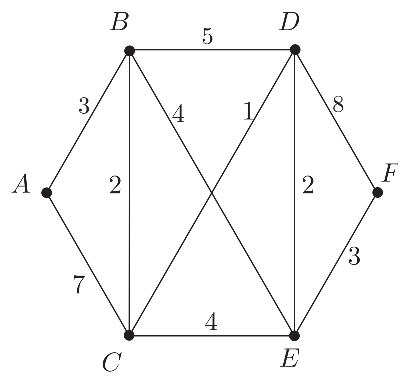
   If there are any piles that still contain 2 or more pennies, chose one of them at random and repeat the two steps. Keep doing this until you end up with $n$ piles each containing one penny. Sum the products that you computed.

   (a) Prove by induction that the sum you get is $n(n-1)/2$.

   (b) Using Graph Theory show that the sum you get is $n(n-1)/2$.

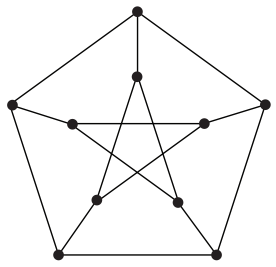
5. Apply Kruskal's Algorithm to construct a minimum-length spanning tree for the graph on the right.

6. Apply Dijkstra's Algorithm to construct a shortest-path spanning tree from the vertex $A$ for the graph on the right.



7. Consider a planar graph which may not be connected, and denote the number of its components by $C$. Prove by mathematical induction on the number $E$ of edges that $V + F = E + C + 1$, where $V$ and $F$ denote the number of vertices and faces, respectively. Deduce from this Euler's Formula.

8. Deduce from Euler's Formula that the Petersen graph shown in the diagram below is non-planar.



Petersen graph

9. Use Kuratowski's Theorem to prove that the Petersen graph is non-planar.

10. What is the maximum number of vertices in a graph of diameter 2 if the degree of each vertex is at most 3?

## 5.9 Questions

1. (Read problem 2 in section 2.1 of *Ecco.*) We have captured several people whom we suspect are part of a spy ring. They are identified as $A$, $B$, $C$, $D$, $E$, $F$, and $G$. After interrogation, $A$ admits to having met the other six. $B$ admits to having met five, $C$ to having met four, $D$ to having met three, $E$ to having met two, $F$ to having met two, and $G$ to having met one.

   None of them would identify whom they knew, and no spy would claim to have met more people than he has actually met. Assume that $F$ is telling the truth and there is only one liar.

   Who is the lying spy, if it is known in addition that the number of acquaintances he gives is 3 less than the true value? Why?

2. Solve Problem #6 of the Omniheurist's Contest in Section 4.2 of *Ecco.*

3. (Read Section 1.3 of *Ecco.*) A substantial collection of jewels is in a chest in one of two underground labyrinths, and they are in a room with an odd number of doors. Each door connects two different rooms. The first labyrinth has two entrance doors and the other has three, and only one of these labyrinths has any rooms with an odd number of doors.

   Assume that the labyrinth with two entrance doors does not contain any room with an odd number of doors. Prove that it is possible to enter this labyrinth by one entrance door and exit by the other.

4. (Read Section 4.4 of *Ecco*.) There are seven code words, denoted by $A$, $B$, $C$, $D$, $E$, $F$, and $G$. All code words have different frequencies: $A$ occurs on the average of 10 times out of 100; $B$, 20 out of 100; $C$, 9 out of 100; $D$, 31 out of 100; $E$, 7 out of 100; $F$, 4 out of 100; and $G$, 19 out of 100. All our messages are to be sent in dots and dashes, but unlike Morse code, which has pauses between letters, we want the code words to go out without pauses.

   Trained people can send dots accurately at the rate of two per second, including the silence before the next dot or dash. Dashes are slower, however, achieving rates of only one per second.

   Suppose that we are not allowed to use "dot dot" as a code word. Design an unambiguous code such that an average message of 100 code words takes no more than 200 seconds.

5. Solve Problem #3 of the Omniheurist's Contest in Section 3.2 of *Ecco*.

6. (Read Section 4.6 of *Ecco*.) The temple that Chief Inspector Singh searched had no windows and only one entrance. Of its rooms, all but one connected to one other room or to three others. The last room connected to two other rooms. There were no doors of any kind between rooms. There is only one way to walk from any room to any other in the temple.

   No traps or barriers were permitted between rooms. The temple is small enough that moving from room to room takes almost no time. Doing a thorough search of a room takes a keeper 20 minutes. At no time should the tiger have a free path to the entrance.

   Design the layout of a temple with the same properties as stated by Chief Inspector Singh, except for the number of rooms, and that an escaped tiger inside can be trapped by two keepers in 2 hours and 20 minutes. What is the maximum number of rooms?

7. During wartime, the queen and her prime minister live in fortified rooms. The queen has 15 rooms and the prime minister has 5, for a total of 20 rooms. Some of the rooms are connected by passageways. No two rooms are connected by more than one passageway.

   For each two of the queen's rooms, there is a unique path of passageways from one to the other.

   The combined total number of passageways for the queen's rooms and the prime minister's rooms is 25. Prove that there is a passageway between one of the queen's rooms and one of the prime minister's rooms.

8. A graph has 11 vertices labelled $A$ to $K$, and the following 18 edges with their lengths given:

$$AB(8), AC(5), AD(7), BE(6), BF(2), CE(4), CF(5), DF(4), DG(2),$$

$$EH(4), FH(4), FI(2), FJ(4), GI(2), GJ(4), HK(4), IK(5), JK(4).$$

   Apply Kruskal's Algorithm to construct a minimum-length spanning tree.

9. Apply Dijkstra's Algorithm to construct a shortest-path spanning tree from the vertex $A$ for the graph in Problem 8.

10. Let $G$ be a graph whose vertices correspond to the bit-strings of length $n$,

$$a = a_1 a_2 \cdots a_n$$

where $a_i = 0$ or 1, and whose edges are formed by joining those bit-strings which differ in exactly two places.

    (a) Show that $G$ is regular, that is, every vertex has the same degree, and find the degree of each vertex.

    (b) Find a necessary and sufficient condition that there exist a path joining two vertices $a = a_1 a_2 \cdots a_n$ and $b = b_1 b_2 \cdots b_n$ in $G$.

    (c) Find the number of connected components of $G$.

11. Solve Problem 2 in Section 2.3 of *Ecco* if conveyers are needed from the loading docks to $B$ and $F$ but not to $A$.

12. A graph consists of 7 vertices $V_1, V_2, \ldots, V_7$. Two vertices $V_i$ and $V_j$ are joined by an edge if and only if the absolute difference between $i$ and $j$ is neither 3 nor 4. Is this graph planar? If so, draw it without crossing edges. If not, explain why not.

13. Solve the Problem in Section 3.3 of *Ecco* if there are 7 rooms with 3 doors each, 8 rooms with 1 door each, and a plot of land 100 feet by 100 feet.

14. Prove that a tree has at most two centres.

15. Let $G = (V, E)$ be a graph with vertex set $V$ and edge set $E$, and let $p$ be the number of vertices in $G$, and let $q$ be the number of edges. The average degree of the vertices in $G$ is defined to be

$$A(G) = \frac{1}{p} \sum_{v \in V} \deg(v).$$

If $G$ is a connected graph, what can you say about $G$ if

    (a) $A(G) > 2$?

    (b) $A(G) = 2$?

    (c) $A(G) < 2$?

Draw a few pictures before committing yourself!!!

# 6 Digraph Theory

**Read Sections 1.1, 1.5, 1.6, 5.6, 6.4, and 6.5 of *Ecco*.**

## 6.1 Basic properties

The Patagonian voting practices grew out of an incident that happened some time ago. There were three candidates in an up-coming election, Jones, Kelly, and Levi. In a survey before the election a majority indicated that they preferred Levi to Jones, and a week later another survey indicated that the voters also preferred Levi to Kelly. However, when the election was held the results were as follows:

| | |
|---|---|
| Jones | 6310 votes |
| Levi | 4300 votes |
| Kelly | 2030 votes |

How could this happen? Well, it turned out that the supporters of Jones and Kelly were extremely opposed to the other candidate. When faced with a choice between Levi and Kelly, the Jones supporters would all vote for Levi, and similarly, the Kelly supporters all preferred Levi to Jones.

A candidate who would win a one-to-one contest against each of the others is called a ***Condorcet candidate***. The word *Condorcet* (pronounced "condor-SAY") comes from the Marquis de Condorcet in the eighteenth century who put it this way: *Whatever judging method is used, if a candidate is preferred in all one-on-one pairings with other candidates then that candidate, in all fairness, should win.* In order to make sure that a Condorcet candidate wins, the Patagonian parliament passed legislation so that all elections would be decided by a sequence of one-on-one competitions. In other words, if we have candidates $A$, $B$, $C$, and $D$, then $A$ would be pitted against $B$. The winner would take on $C$, and the winner of that would take on $D$.

We can indicate voter preferences by a graph-like structure. For the Jones–Kelly–Levi election, the one-on-one voter preferences could be displayed by Figure 6.1

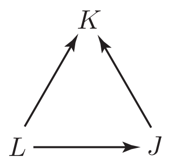


Figure 6.1.

The arrows indicate that voters prefer J to K, prefer L to J, and prefer L to K. The figure is an example of a directed graph. Formally, a ***directed graph*** or ***digraph*** $G$ is an ordered pair $(V, E)$ where $V$ is a set of elements called ***vertices*** and $E$ is a set of *ordered pairs* $(u, v)$ of elements in $V$ called ***arcs***. Informally, a digraph is just a graph or multigraph with arrows put on the edges. Thus each digraph $G$ has an underlying graph $G'$ obtained from $G$ by ignoring the orientations of the arcs and converting them into edges.

The concepts of incidence, adjacency and subgraphs are defined in the same way as for graphs. An arc $a = (u, v)$ is said ***to go from** $u$ **to** $v$*. A ***directed path*** is a sequence $v_0, a_1, v_1, a_2, v_2, \ldots, a_n, v_n$ such that for $0 \leq i \leq n$, $v_i$ are distinct vertices (except perhaps that $v_0$ and $v_n$ may be the same), and such that for $1 \leq i \leq n$, $a_i$ is an arc going from $v_{i-1}$ to $v_i$. We say that this path takes us from $v_0$ to $v_n$. A ***directed cycle*** is defined just like a directed path except that $v_0 = v_n$. The notions of a ***directed walk***, ***directed trail*** and ***directed circuit*** are also related in a similar way to the corresponding notions for graphs.

A digraph is said to be ***weakly connected*** if the underlying graph is connected. It is said to be ***strongly connected*** if for any ordered pair $(u, v)$ of vertices there is a path which takes us from $u$ to $v$. The ***in-degree*** of a vertex is the number of arcs going to it, and the ***out-degree*** of a vertex is the number of arcs going from it.

**Theorem 6.1.1 (The Parity Theorem for Digraphs).** *The sum of the in-degrees of all vertices of a digraph is equal to the sum of the out-degrees of all vertices, and this common value equals the number of arcs.*

As a formula, this becomes:

$$\sum_{v \in V} \deg^-(v) = \sum_{v \in V} \deg^+(v) = q,$$

where $q$ is the number of edges, $V$ is the vertex set, and $\deg^-(v)$ and $\deg^+(v)$ are the in-degree and out-degree of a vertex $v$.

### 6.1.1 Tournaments

A ***tournament*** is a digraph whose underlying multigraph is a complete simple graph. If we consider the vertices as teams in a round-robin tournament, then the arcs may be considered as the result of the game between two teams. For instance, the preference chart on the Amazon proposals in Section 1.1 of *Ecco* generates the tournament on the right.

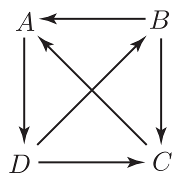

$$\begin{tikzcd} A \arrow[d] & B \arrow[l] \arrow[d] \\ D \arrow[r] \arrow[ur] & C \arrow[ul] \end{tikzcd}$$

Figure 6.2.

To generate the tournament from the voting block preference charts, you have to check all one-on-one competitions. For example, the Amazon proposals divided the 100 members of the Patagonian parliament into the following voting blocks:

| preference chart | size of voting block |
|---|---|
| $(C, A, D, B)$ | 17 |
| $(A, B, D, C)$ | 32 |
| $(D, B, C, A)$ | 34 |
| $(B, A, C, D)$ | 17 |

The notation $(W, X, Y, Z)$ for a voting block means that everyone in that voting block would vote for $W$ over $X$, $Y$, or $Z$; would vote for $X$ over $Y$ or $Z$; and would vote for $Y$ over $Z$. In the preference charts given in the table, two voting blocks prefer $B$ to $A$, and these blocks together comprise 51 of the 100 voters. So in a head-to-head election between $A$ and $B$, the total votes for $B$ would be 51 and the total votes for $A$, 49, so $B$ wins. In a one-on-one competition between $B$ and $C$, the total votes for $B$ are $32 + 34 + 17$ (from the last three voting blocks), and $B$ wins over $C$. Continuing in this way, we generate the tournament in Figure 6.2

**Example 6.1.2.** *Find another preference chart which generates the same tournament, but using the minimum number of voting blocks.*

*Solution.* Since $D$ beats $B$, $B$ beats $C$, $C$ beats $A$ and $A$ beats $D$, one voting block is not sufficient. If one of two voting blocks is larger than the other, then the smaller one may as well be eliminated. If they are of the same size, it will not produce clear-cut winners in all cases. Hence we need at least three voting blocks. This can be accomplished with the three voting blocks $(D, B, C, A)$, $(A, D, B, C)$, and $(B, C, A, D)$. The actual sizes of the voting blocks does not matter as long as each is strictly less than half of the voting population. □

A directed path or cycle that contains all vertices of the digraph is said to ***span*** the digraph. Problem 3 of Section 1.2 in *Ecco* asked for a proof that any of the Amazon proposals can win in a sequence of one-on-one elections. Examination of the tournament reveals the existence of a spanning directed cycle, namely, $A \to D \to B \to C \to A$, from which the desired result follows immediately. For example, if we want $C$ to win, the sequence of one-on-one elections should be:

| | |
|---|---|
| first election: | $B$ versus $D$ |
| second election: | winner versus $A$ |
| third election: | winner versus $C$ |

A spanning directed cycle, such as the one in Figure 6.2, is called a ***directed Hamiltonian cycle***. Clearly, if a tournament has a directed Hamiltonian cycle, it is strongly connected. It turns out that the converse is also true, but first we prove a related result about ***Hamiltonian paths***, which are directed paths (not necessarily cycles) that span the digraph.

**Theorem 6.1.3 (Redei's Theorem).** *Every tournament has a directed Hamiltonian path.*

*Proof.* Since the underlying graph is complete, if $u$ and $v$ are two vertices, there is an arc $(u, v)$ or an arc $(v, u)$, so the tournament has a directed path of length 2. Let $v_1, v_2, \ldots v_k$ be a directed path. We will show that if there are some vertices of the tournament that are not on this path, they can be added to the path one-by-one. This will prove the theorem.

Suppose that $v$ is not on the path. There are arcs between $v$ and each of the vertices on the path. If there is an arc $(v, v_1)$, add $v$ to the beginning of the path. If this is not the case but there is an arc $(v_k, v)$, add $v$ to the end of the path. If neither is possible, then there must be arcs $(v_1, v)$ and $(v, v_k)$ as shown in Figure 6.3.

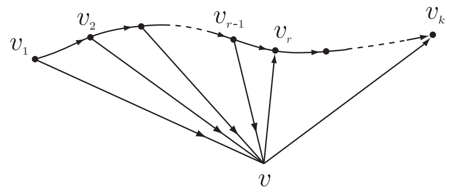
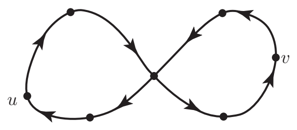


Figure 6.3.

Now consider in order the vertices $v_2, v_3, \ldots,$ and let $v_r$ be the first vertex with an arc from $v$ to $v_r$ (which includes the possibility that $v_r = v_k$). The situation is depicted in the figure. Then we can insert $v$ into the path between $v_{r-1}$ and $v_r$. ☐

**Question 6.1.4.** *If a digraph has a directed path from $u$ to $v$ and a directed path from $v$ to $u$, is it necessarily the case that there there is a directed cycle containing $u$ and $v$?*

The answer is no, as shown by the graph below.



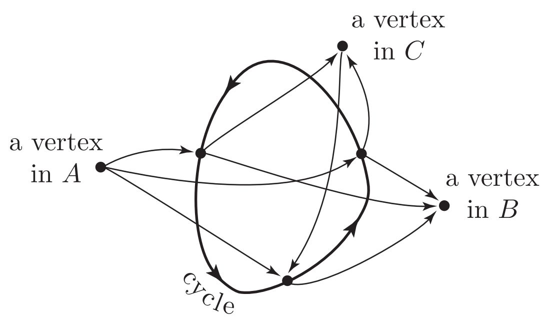
**Example 6.1.5.** *Show that if a digraph has a directed path from $u$ to a different vertex $v$ and a directed path from $v$ to $u$, then there is a cycle containing $v$.*

*Solution.* If the directed paths from $u$ to $v$ and from $v$ to $u$ have no other vertices in common, then we have the desired cycle. If this is not the case, then one or more vertices of the directed path from $u$ to $v$ are also on the directed path from $v$ to $u$. Label the directed path from $u$ to $v$ as $u_1(= u), u_2, u_3, \ldots, u_k(= v)$. Let $u_r$ be the vertex with the highest index that is also on the directed path from $v$ to $u$. Now $u_r$ is not the same as $u_k$. We have a directed path from $u_r$ to $v$ and a directed path from $v$ to $u_k$ and these two paths have no other vertices in common. Thus we have found a directed cycle containing $v$. ☐

**Theorem 6.1.6 (Camion-Moon Theorem).** *Every strongly connected tournament has a Hamiltonian cycle.*

*Proof.* Pick any two vertices $u$ and $v$. Since the tournament is strongly connected, there is a path from $u$ to $v$ and a path from $v$ to $u$. So there is a directed cycle containing $v$. If this cycle contains all of the vertices of the tournament, we are finished. If it does not, we will show that we can add more vertices to the directed cycle.

Examine the vertices that are not in the cycle. Each vertex not in the cycle is joined to every vertex on the cycle by arcs. We divide the vertices not on the cycle into three sets: A vertex belongs to set $A$ if the arcs between the vertex and the cycle all lead *to* the cycle. A vertex belongs to set $B$ if the arcs between the vertex and the cycle all lead *to* the vertex. All remaining vertices not on the cycle belong to set $C$. (Some of the sets may be empty.)



Suppose that there is a vertex $v$ in set C. Just as in the proof of Redei's Theorem, there must be two consecutive vertices, say $u$ and $w$, on the directed cycle such that there is an arc $(u, v)$ and an arc $(v, w)$. Then we can replace the arc $(u, w)$ with the directed path $(u, v, w)$, and thereby insert $v$ into the cycle.

Suppose that the set $C$ is empty. Then both $A$ and $B$ are empty, or both are nonempty. (If set $A$ is not empty and $B$ were empty, there would be no path leading from a vertex on the cycle to any vertex in $A$, so the tournament would not be strongly connected.) So we may assume that both $A$ and $B$ are nonempty. There are arcs between the vertices of $A$ and $B$, and at least one of these arcs must lead from a vertex in $B$ to a vertex in $A$ (else the tournament would not be strongly connected). Let us suppose that there is an arc from $v$ in $B$ to $z$ in $A$. Let $u$ and $w$ be any two consecutive vertices on the directed cycle. Then, since $v$ is in $B$, there is an arc $(u, v)$. Similarly there is an arc $(z, w)$. Replace the arc $(u, w)$ with the path $(u, v, z, w)$, thereby inserting two of the remaining vertices into the cycle.

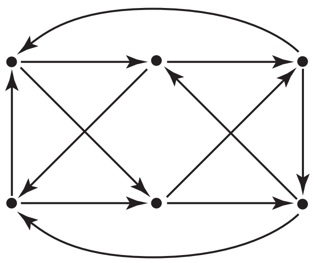
This completes the proof of the Camion-Moon Theorem. □

A digraph is also useful in representing the situation in Section 5.6 of *Ecco*. Here, the vertices are the control stations, and the arcs are the one-way communication units. Clearly, a strongly connected digraph is desirable. Moreover, we would like the digraph to have as small a diameter as possible, where the concepts of radius, diameter and centre are as in ordinary graphs except for taking into consideration the orientations of the arcs (see page 103 for more infomation about these concepts).

**Example 6.1.7.** *In a certain digraph of diameter 2, each vertex has in-degree 2 and out-degree 2. What is the maximum number of vertices in such a digraph?*

*Solution.* Consider an arbitrary vertex $r$. We can reach two other vertices $u$ and $v$ in one step, and four other vertices in two steps, say $w$ and $x$ via $u$, and $y$ and $z$ via $v$. Hence the digraph can have at most 7 vertices.

Suppose it has exactly 7. Then $u$ must reach $r$, $v$, $y$ and $z$ in two steps via $w$ and $x$. We may assume that there is an arc going from $w$ to $r$. Consider now $w$. We can reach $u$ and $v$ via $r$ in two steps, and we still have to get to $x$, $y$ and $z$. We cannot have an arc going from $w$ to $x$, as it would not be possible to go from $u$ to every other vertex in two steps. Hence we may assume that there is an arc going from $w$ to $y$, and arcs from $y$ to $x$ and $z$. However, it is not possible to go from $v$ to every other vertex in two steps. So we cannot have seven vertices, but we can have six as the following digraph shows: □



There is another consideration, arising from the possibility of a control station being incapacitated. In order to still have a strongly connected digraph after the removal of any vertices, along with its arcs, we need the underlying graph to be more than just connected.

An ***articulation set*** is a set of *vertices* such that their removal, along with incident edges, leaves behind either a single vertex or a disconnected graph. Clearly, a communication network should not have an articulation set of size 1.

An articulation set of size 2 may also cause problems. Suppose $\{u, v\}$ is such a set, and the removal of $u$ and $v$ leaves behind a disconnected subgraph $H$. Let $A$ be one of its components. In the digraph, suppose all arcs between $u$ and $A$ go from $u$ to $A$. Then removal of $v$ will leave behind a digraph which is not strongly connected.

A ***cut set*** is a set of *edges* such that their removal leaves behind a disconnected graph. A cut set of size 1 is called a ***bridge***. Clearly, either vertex incident with a bridge is an articulation set of size 1.

Suppose we have a cut set of size 2, consisting of the edges $e$ and $f$, such that their removal leaves behind a disconnected subgraph $H$. Let $A$ be one of its components. If the digraph is to be strongly connected, we must orient one of $e$ and $f$ towards $A$ and the other away from it. Now the removal of one of the vertices incident with $e$ or $f$ will leave behind a digraph which is not strongly connected.

## 6.2 Critical Paths

Turning now to Section 1.6 of *Ecco*, where a project consists of a number of tasks. Each task $K$ has a duration denoted by $d(K)$. Moreover, some tasks cannot start until others have been completed. Of course, we cannot have cyclic dependence.

The objective is to complete the project in the shortest possible time. To this end, we identify certain individual tasks as ***critical tasks***, in that any delay in performing such a task will lead to a delay of the project. Critical tasks cannot exist in isolation, but must congregate into ***critical paths***. We now give an algorithm which finds the critical paths.

***The Critical Path Algorithm***

First, we list the tasks so that if $L$ depends on the completion of $K$, then $K$ is listed before $L$, though not necessarily immediately. We add a fictitious task $X$ at the end to signify the completion of the project, and make it dependent on tasks on which no other tasks depend.

For each task $K$, we compute its ***earliest starting time*** $t(K)$ as follows. If $K$ does not depend on the completion of any other task, we set $t(K) = 0$. Otherwise, we take $t(K) = \max\{t(J) + d(J)\}$, the maximum taken over all tasks $J$ on which $K$ depends. Note that $t(X)$ is the minimum time for the completion of the project.

Next, we compute the ***latest starting time*** $T(K)$ of each task $K$ as follows. We work our way backwards along the list and set $T(X) = t(X)$. For $K \neq X$, we take $T(K) = \min\{T(L) - d(K)\}$, the minimum taken over all tasks $L$ which depends on $K$.

Finally, we compute the ***slack*** $s(K)$ of a task $K$ as the difference between its latest starting time and its earliest starting time. In other words, $s(K) = T(K) - t(K)$. If $s(K) = 0$ and $K \neq X$, then $K$ must be a *critical task*.

**Example 6.2.1.** *Apply the critical-path algorithm to Mr. Henderson's project in Section 1.6.*

*Solution.* The following chart shows the listing of the tasks, along with their durations, and the computations of their earliest starting time, latest starting time and slack.

| $K$ | $B$ | $C$ | $F_1$ | $A$ | $F_2$ | $D_1$ | $D_2$ | $E$ | $X$ |
|---|---|---|---|---|---|---|---|---|---|
| $d(K)$ | 4 | 4 | 1.5 | 2 | 1.5 | 2 | 2 | 3 | 0 |
| $t(K)$ | 0 | 0 | 0 | 4 | 1.5 | 1.5 | 3.5 | 3.5 | 6.5 |
| $T(K)$ | 0.5 | 2.5 | 0 | 4.5 | 5 | 1.5 | 4.5 | 3.5 | 6.5 |
| s $s(K)$ | 0.5 | 2.5 | 0 | 0.5 | 3.5 | 0 | 1 | 0 | 0 |

In computing $t(K)$, note that $X$ depends on $A, C, F_2, D_2$ and $E$. We have $t(A) + d(A) = 6$, $t(C) + d(C) = 4$, $t(F_2) + d(F_2) = 3$, $t(D_2) + d(D_2) = 5.5$ and $t(E) = d(E) = 6.5$. Hence $t(X) = 6.5$. In computing $T$, note that $D_2$ and $E$ depend on $D_1$, and $F_2$ and $D_1$ depend on $F_1$. We have $T(D_2) - d(D_1) = 2.5$ and $T(E) - d(D_1) = 1.5$. Hence $T(D_1) = 1.5$. Similarly, since $T(F_2) - d(F_1) = 3.5$ and $T(D_1) - d(F_1) = 0$, we have $T(F_1) = 0$. The critical tasks are $F_1, D_1$ and $E$, and they form a critical path leading to $X$. □

## 6.3 Transportation Networks

In Section 1.5 of *Ecco*, we are dealing with a ***transportation network***. This is a directed graph with the following properties.

1. There are no directed cycles.
2. There is a unique vertex with in-degree 0, called the ***source***.
3. There is a unique vertex with out-degree 0, called the ***sink***.
4. Each arc $a$ is associated with a positive real number $c(a)$, called its ***capacity***.

A ***flow*** in a transportation network is a function $f$ from the arcs to the non-negative real numbers with the following properties.

1. For every arc $a, f(a) \leq c(a)$.
2. For every vertex other than the source or the sink, the sum of $f(a)$ on all arcs $a$ leading into this vertex is equal to the sum of $f(a)$ on all arcs $a$ leading from this vertex.

Clearly, the sum of $f(a)$ on all arcs $a$ leading from the source will be equal to the sum of $f(a)$ on all arcs $a$ leading into the sink. This common value is called the ***value*** of the flow. The flow with the highest value is called a ***max-flow***.

A ***cut*** in a transportation network is a partition of the vertices into two sets, one containing the source and the other containing the sink. (Note that this is a specialized version of the notion of a *cut set* defined on page 98.) The ***value*** of a cut is the sum of $c(a)$ on all arcs leading from the first set into the second. The cut with the lowest value is called a ***min-cut***.

It is easy to see that the value of *any* flow is less than or equal to the value of *any* cut. If we can find a flow and a cut with the *same* value, then the flow must be a max-flow and the cut must be a min-cut.

**Theorem 6.3.1 (The Max-flow Min-cut Theorem).** *If the capacities of all the arcs in a transportation network are integral, then both a max-flow and a min-cut exist.*

This theorem is a consequence of the Ford-Fulkerson Algorithm, which we present below. The justification that it always provides a max-flow and min-cut is given following an example of its application.

***The Ford-Fulkerson Algorithm***

List the vertices so that $u$ precedes $v$, not necessarily immediately, if there is an arc leading from $u$ into $v$. This is possible because of the absence of directed cycles. The source is always listed first and the sink last. Label the source $(-,\infty)$. For each subsequent vertex $v$, label it $(-,0)$ if no arcs lead into it, and delete all arcs leading from it. Otherwise, label it $(u,n)$ where $u$ is the first vertex on the list with an arc leading from it into $v$, and $n$ is the capacity of this arc.

If the sink is labelled anything other than $(-,0)$, we can generate a flow by backtracking through the labels along a path to the source. The value of the flow is the minimum value of the current capacities of all the arcs along this path. We then reduce all capacities along the path by this amount. If the capacity of an arc is reduced to 0, it is deleted. If all arcs leading into a vertex are deleted, then all arcs leading from it are also deleted. The algorithm terminates when the sink is labelled $(-,0)$. The flow is the combination of all those generated. The corresponding cut puts all vertices labelled $(-,0)$ in the second set and the remaining vertices in the first.

**Example 6.3.2.** *Solve the shipper's problem in Section 1.5 of* Ecco *by the Ford-Fulkerson Algorithm.*

*Solution.* An application of the Ford-Fulkerson Algorithm yields the following chart:

| $H$ | $L$ | $P$ | $F$ | $R$ | $W$ | $M$ | Path | Flow |
|---|---|---|---|---|---|---|---|---|
| $-,\infty$ | $H,10$ | $H,11$ | $H,3$ | $H,3$ | $L,8$ | $P,2$ | $P$ | 2 |
| $-,\infty$ | $H,10$ | $H,9$ | $H,3$ | $H,3$ | $L,8$ | $F,8$ | $F$ | 3 |
| $-,\infty$ | $H,10$ | $H,9$ | $P,10$ | $H,3$ | $L,8$ | $F,5$ | $PF$ | 5 |
| $-,\infty$ | $H,10$ | $H,4$ | $P,5$ | $H,3$ | $L,8$ | $R,3$ | $R$ | 3 |
| $-,\infty$ | $H,10$ | $H,4$ | $P,5$ | $-,0$ | $L,8$ | $W,7$ | $LW$ | 7 |
| $-,\infty$ | $H,3$ | $H,4$ | $P,5$ | $-,0$ | $L,1$ | $-,0$ | | |

The max-flow $f$ of value 20 is defined by $f(HL)=7,\ f(HP)=7,\ f(HF)=8,\ f(HR)=3,$ $f(LW)=7,\ f(PM)=2,\ f(PF)=5,\ f(FM)=8,\ f(RM)=3$ and $f(WM)=7$. The corresponding min-cut has $H,\ L,\ P,\ F$ and $W$ on one side and $R$ and $M$ on the other. Its value is $c(HR)+c(PM)+c(FM)+c(WM)=3+2+8+7=20$. □

***Justification of the Ford-Fulkerson Algorithm***

It is easy to see that if the algorithm terminates, the flow generated and the resulting cut must have the same value. Hence the former is a max-flow and the latter is a min-cut. Since all capacities are integral and there are only a finite number of them, the algorithm must terminate.

### 6.3.1 Orienting a graph

Let $G$ be a connected graph. An ***orientation*** of $G$ is a conversion of $G$ into a directed graph by making each edge into an arc. If $G$ has a bridge, it is easy to see that no orientation of $G$ can be strongly connected. On the other hand, if $G$ has no bridge, we will prove that it has a strongly connected orientation.

We will need the concept of a ***rooted directed spanning tree***. It is a spanning tree with one of the vertices designated as the root, and all arcs are oriented away from it. Such a tree will play a central role in *Robbins' Algorithm* which is an algorithm for finding a strongly connected orientation for a connected bridgeless graph.

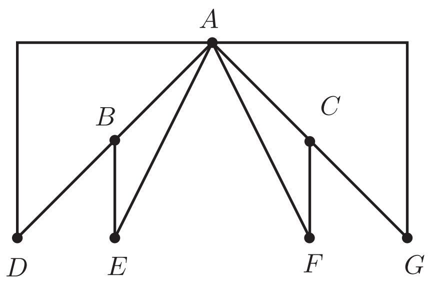
***Robbins' Algorithm***

List the vertices of the given graph in any order. Label the first vertex 1. Make it the root of a rooted directed spanning tree $T$ which we will be constructing. Let the active vertex be labelled $k$, where k is some positive integer.

1. If all vertices have been labelled, the iteration is terminated.
2. If there are still unlabelled vertices but none of them is joined to the active vertex, we must have $k > 0$ since the given graph is connected. Make $k - 1$ the active vertex instead and continue with the iteration.
3. If some unlabelled vertex is connected to the active vertex, label it $\ell$ where $\ell$ is the smallest positive integer not used in any label. Put into $T$ the arc going from the active vertex to it, and make it the active vertex. Continue with the iteration.

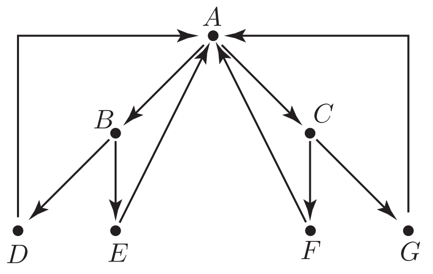
Each vertex is now labelled with some positive integer and $T$ is a directed spanning tree rooted at the vertex labelled 0. For any edge which is not in $T$, direct it from the vertex with the higher label to the lower. This completes the construction of the orientation.

**Example 6.3.3.** *Apply Robbins' Algorithm to the following graph.*



*Solution.* We list the vertices in alphabetical order. Label vertex $A$ with a 1 and make it the active vertex. Since $B$ is connected to it, label $B$ with a 2 and make it the active vertex.

Similarly, $D$ is labelled with a 3 and made the active vertex. Since there are no unlabelled vertices connected to $D$, we bring $B$ back as the active vertex. Label $E$ with 4, and since no unlabelled vertices are connected to it, the active vertex is backed up through $B$ to $A$. Label $C$ with 5, $F$ with 6 and $G$ with 7 as before. This completes the construction of $T$ and results in the following orientation.



□

***Justification of Robbins' Algorithm.***

Clearly, from the root 0, we can reach any other vertex $k$ along the arcs of $T$. We now prove by mathematical induction on $k$ that we can also reach 0 from $k$. It will then follow that the orientation is indeed strongly connected. The result is trivial for $k = 0$. Suppose it holds for all $k, 0 \leq k \leq \ell - 1$ for some positive integer $\ell$. Let $T_\ell$ be the directed subtree of $T$ rooted at $\ell$. Then its vertices are $i$, $\ell \leq i \leq j$ for some $j \geq \ell$. Since the arc in $T$ leading to $\ell$ was not a bridge in the given graph, there is an edge between some $i$ in $T_\ell$ and some $t$ not in $T_\ell$. We must have $t \leq \ell - 1$ as otherwise $t$ would have been in $T_\ell$. Hence this edge is directed from $i$ to $t$. We can reach $i$ from $\ell$ along the arcs in $T$, and then go from $i$ to $t$. By the induction hypothesis, we can reach 0 from $t$.

## 6.4 Centre, radius, and diameter of digraphs

The strongly connected orientation produced by Robbins' Algorithm is often not the most efficient way of converting into one-way streets. In a digraph, the ***distance*** from a vertex $u$ to another vertex $v$ is defined to be the number of arcs on the shortest path joining $u$ and $v$. The ***radius*** of a vertex and the ***radius***, ***diameter*** and ***centre(s)*** of the digraph are then defined in the same way as in an undirected graph.

**Example 6.4.1.** *Find the radius of each vertex of the digraph obtained in Example 6.3.3, and determine the radius, diameter, and centres of the digraph.*

*Solution.* This task can be accomplished by checking the distances from each vertex to all others. From this we can obtain the radius of each vertex as in the following chart.

| | $A$ | $B$ | $C$ | $D$ | $E$ | $F$ | $G$ | Radius |
|---|---|---|---|---|---|---|---|---|
| $A$ | 0 | 1 | 1 | 2 | 2 | 2 | 2 | 2 |
| $B$ | 2 | 0 | 3 | 1 | 1 | 4 | 4 | 4 |
| $C$ | 2 | 3 | 0 | 4 | 4 | 1 | 1 | 4 |
| $D$ | 1 | 2 | 2 | 0 | 3 | 3 | 3 | 3 |
| $E$ | 1 | 2 | 2 | 3 | 0 | 3 | 3 | 3 |
| $F$ | 1 | 2 | 2 | 3 | 3 | 0 | 3 | 3 |
| $G$ | 1 | 2 | 2 | 3 | 3 | 3 | 0 | 3 |

The diameter of the digraph is 4, the radius 2 and the centre $A$. □

## 6.5 Exercises

1. Among six candidates, the electorate is divided into three voting blocks:

$$(A, B, C, D, E, F), \quad (C, F, D, A, E, B), \quad \text{and} \quad (F, E, A, C, B, D),$$

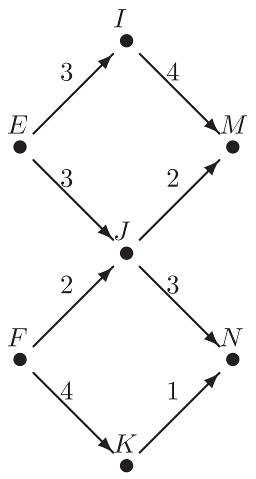
each numbering less than half. Determine all candidates who can emerge as the eventual winner in a sequence of five one-on-one elections, in each of which the loser is eliminated.

2. Prove that the radius of a tournament is at most 2.

3. Prove that if the radius of a tournament is 2, then the minimum number of its centres is three.

4. A job consists of 6 tasks labelled $A$ to $F$. Given below for each are its duration in weeks, as well as other tasks that must have already been completed: $A(3;-)$, $B(3;A)$, $C(4;-)$, $D(1;B)$, $E(2;B,C)$ and $F(1;A,E)$. Apply the critical-path algorithm to compute the slack time of each task and find the critical path.

5. Factory $E$ can produce 5 units of goo while factory $F$ can produce 6. Market $M$ has ordered 6 units of goo while market $N$ has ordered 5. The transportation network is shown in the diagram below, where the numbers denote the capacities of the arcs in units of goo. To what extent can the orders be filled, with priority going to $M$?



6. A graph has 7 vertices labelled $A$ to $G$, and the following 12 edges: $AB, AD, AF, BC$, $BD, BE, BF, BG, CE, CG, DE$ and $FG$. Apply Robbins' Algorithm to convert it into a strongly connected directed graph, with $A$ as the root.

7. In your solution to Exercise 6, find the radius, diameter and centres of the resulting digraph.

## 6.6 Questions

1. In Section 1.1 of *Ecco*, if the opposition chooses the first pair of Amazon proposals and Libretti the second, can he ensure that $C$ or $A$ will win? If so, show how. If not, explain why not.

2. For each of the tournaments in the diagram below, construct a preference chart which may generate it. Use as few voting blocks as possible.

(a)
$$\begin{tikzcd}
A \arrow[r] \arrow[d] \arrow[dr] & B \arrow[d] \arrow[dl] \\
D & C \arrow[l]
\end{tikzcd}$$

(b)
$$\begin{tikzcd}
A \arrow[r] \arrow[d] \arrow[dr] & B \arrow[d] \\
D \arrow[ur] & C \arrow[l]
\end{tikzcd}$$

(c)
$$\begin{tikzcd}
A \arrow[r] \arrow[d] & B \arrow[d] \arrow[dl] \\
D & C \arrow[l] \arrow[ul]
\end{tikzcd}$$

3. The Floridian parliament has to choose its next prime minister. There are five candidates, namely:

   Axel, Boris, Claudia, Dieter, Erica

   The house is split into 3 different voting blocks, each with 30 voters. The preferences are shown below.

   $$(B, A, D, E, C)$$
   $$(A, E, C, D, B)$$
   $$(D, C, B, E, A)$$

   Here $(U, V, W, X, Y)$ means that a voting block prefers $U$ to $V$ to $W$ to $X$ to $Y$. The Floridian system uses a sequence of one-on-one elections with the winner of one election being a candidate in the next election. The sequence is set by the chief justice.

   (a) Draw the tournament which is generated by these preferences charts.

   (b) The chief justice wants Claudia to win. What sequence should he use.

4. In Section 5.6 of *Ecco*, suppose that none of the 15 stations can be incapacitated. Solve General Lange's problem, but with condition only 22 one-way communication units can be used, and that each subordinate station must be able to receive a message from and send a message to the command post within three minutes.

5. In General Lange's problem, suppose each control station has 3 communication units which work both ways. What is the maximum number of control stations so that any two can reach each other within two minutes?

6. In Section 1.6 of *Ecco*, if Mr. Henderson is only willing to spend 10 million dollars, by how much can the project be shortened? What if he is willing to spend 20 million dollars?

7. A job consists of 10 tasks labelled $A$ to $J$. Given below for each are its duration in weeks, as well as other tasks that must have already been completed: $A(7;-)$, $B(3;A)$, $C(8;A)$, $D(1;B)$, $E(2;C,D)$, $F(1;C,D)$, $G(1;C,D)$, $H(2;F)$, $I(2;H)$, and $J(1;E,G,I)$. Apply the critical-path algorithm to compute the slack time of each task and find the critical path.

8. In Section 1.5 of *Ecco*, suppose the shipper may charter any number of planes with unlimited capacity, but no two can stop at the same city. To which existing routes should they be added in order to increase the total shipment amount by as much as possible? How much is that?

9. A transportation network consists of 6 vertices labelled $A$ to $F$, and the following 10 arcs with their capacities given in the brackets: $AB(3)$, $AC(7)$, $BC(2)$, $BD(5)$, $BE(4)$, $CD(1)$, $CE(4)$, $DE(2)$, $DF(8)$, and $EF(3)$. Apply the Ford-Fulkerson Algorithm to find a maximum flow from the source $A$ to the sink $F$ and find the corresponding minimum cut.

10. In Problem 2 of Section 6.4, is Dr. Ecco's solution the only possible one, apart from the one obtained by reversing the orientation of every arc? If so, prove it. If not, find another.

11. In Section 6.5 of *Ecco*, if only 100 miles of country road are to be added, must it all be between $B$ and $D$? If so, explain why? If not, show where else it can be added.

12. A graph has 11 vertices labelled $A$ to $K$, and the following 18 edges: $AB, AC, AD, BE$, $BF$, $CE$, $CF$, $DF$, $DG$, $EH$, $FH$, $FI$, $FJ$, $GI$, $GJ$, $HK$, $IK$ and $JK$. Apply Robbins' Algorithm to convert it into a strongly connected directed graph, with $A$ as the root.

13. In your solution to Question 12, find the radius, diameter and centres of the resulting digraph.

# 7 Miscellaneous Topics

**Read Sections 1.4, 2.2, 3.4, 3.5, 3.6, 4.1, 4.5, 5.3, and 5.8 of *Ecco*.**

## 7.1 Parallel sorting

Before applying Kruskal's Algorithm to find a minimum spanning tree for a connected graph, we have to rank the edges in non-decreasing order of their lengths. There are many other similar problems in real life. For instance, a company handles many files which may arrive in random order, and it is necessary to sort them alphabetically. Moreover, such processes often have to be done over and over again.

In Problem 1 of Section 1.4 in *Ecco*, Coach McGraw has to rank his 8 players from best to worst. Let us recall the problem:

**The Coach's Dilemma.** Eight tennis players have to be completely ranked in twenty hours. Each match takes 1 hour, and there is only one court available. It is assumed that the ranking is transitive, that is, if player **X** beats player **Y** and **Y** beats **Z**, then **X** would beat **Z** if they played each other.

With only 1 court at his disposal, matches are played one at a time. This is known as ***sequential processing***. To get a feel for this problem, let us start with a smaller number of players. Clearly, sorting 1 player requires 0 hours, 2 players need 1 hour, and 3 players require 3 hours. When the number gets larger, a useful strategy is to *divide and conquer*.

Suppose we have 4 players. We first have them go against one another in pairs. In 2 hours, we will have two sorted pairs $(a_1, a_2)$ and $(b_1, b_2)$. We now merge them into a sorted quartet $(c_1, c_2, c_3, c_4)$ as follows. In hours 3 and 4, $a_1$ plays $b_1$ and $a_2$ plays $b_2$. The winner of the first match is $c_1$, and the loser of the second match is $c_4$. If the loser of the first match has already played the winner of the second, the ranking is completed. Otherwise, these two will play in hour 5 to determine $c_2$ and $c_3$. Having solved this problem, we are in a much better position to understand Dr. Ecco's solution to Problem 1.

**Example 7.1.1.** *Sort 5 players in 7 hours using one court.*

*Solution.* In hours 1 and 2, have four players go against one another in pairs. In hour 3, let the two winners go against each other. We now have the following partial ranking, where $X \longrightarrow Y$ indicates that $X$ is better than $Y$.

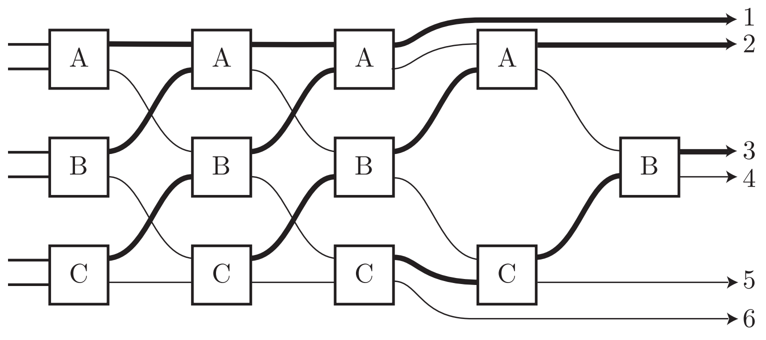
$$\begin{tikzcd}
A \arrow[r] \arrow[d] & B \arrow[r] & C \\
D & & E
\end{tikzcd}$$

In hour 4, E plays B. In hour 5, E plays A after a win over B, or else E plays C after a loss to B. Thus E is incorporated into the main stream which at present consists of A, B and C. In 2 more hours, we can incorporate D as well, using the fact that D has already lost to A. □

In Problem 2, more courts are available, and some comparisons may take place simultaneously. This is known as ***parallel processing***. For instance, it is easy to see that the complete ranking of 4 players may be obtained in only 3 hours if 2 courts are available. This is analogous to the company dealing with files hiring a second sorter to speed up the process. However, it must ensure that the two do not get in each other's way.

**Example 7.1.2.** *Sort 6 players in 5 hours using 3 courts.*

*Solution.* Sometimes it is easier to describe a solution using a picture. In the following chart, the courts are labelled A, B, and C. In each game, the winner proceeds along the heavy line and the loser along the lighter line.



It is clear that the highest ranking player must play in court A by the third round, and if he gets to court A before that, he continues to play there. A similar situation describes the path taken by the lowest ranking player, so that after three rounds we will have discovered the first and sixth player.

The second ranked player can only lose to the 1st ranked player. Consequently, in round 2 the 2nd ranked player can never meet the 1st ranked player in either court B or court C. So in the third round, the second ranked player either loses to the 1st ranked player in court A, or else he wins his match in court B. In either case, he proceeds to court A to play the fourth round. The progress of the 5th ranked player mirrors this, so after 4 rounds, we have ranked 1, 2, 5, and 6. In the fifth round we determine the third and forth ranked players. □

Returning now to Coach McGraw's second problem, it is not immediately clear why Dr. Ecco's solution works, or how he may have thought of it in the first place. Let us go over it more carefully. In hour 1, the players go against one another in pairs, resulting in 4 sorted pairs. In hours 2 and 3, we perform two simultaneous merges of two sorted pairs into a sorted quartet. So far, this is essentially the same approach Dr. Ecco used in solving Problem 1.

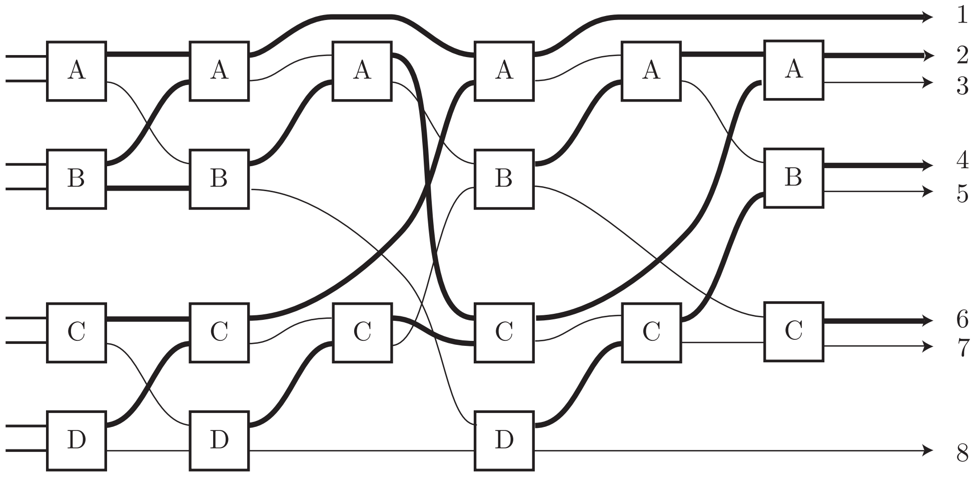
In the final 3 hours, we merge two sorted quartets $(c_1, c_2, c_3, c_4)$ and $(d_1, d_2, d_3, d_4)$ into a sorted octet $(e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8)$ as follows. In hours 4 and 5, we merge $(c_1, c_3)$ with $(d_1, d_3)$ into $(f_1, f_2, f_3, f_4)$ as well as $(c_2, c_4)$ and $(d_1, d_3)$ into $(g_1, g_2, g_3, g_4)$ as before.

Clearly, we have $f_1 = e_1$, as well as $g_4 = e_8$. We claim that $\{f_2, g_1\} = \{e_2, e_3\}$, $\{f_3, g_2\} = \{e_4, e_5\}$ and $\{f_4, g_3\} = \{e_6, e_7\}$, so that at most 1 more hour is needed for completing the ranking. By symmetry, we may assume that $f_1 = c_1$. Then $f_2 = d_1$ or $c_3$ while $g_1 = d_2$ or $c_2$. Now only $c_1$ and possibly $c_2$ can be ahead of $f_2$. Similarly, only $c_1$ and possibly $d_1$ can be ahead of $g_1$. Hence $\{f_2, g_1\} = \{e_2, e_3\}$. By symmetry again, $\{f_4, g_3\} = \{e_6, e_7\}$, so that we have $\{f_3, g_2\} = \{e_4, e_5\}$ as desired.

This method is known as the ***Batcher merge-sort*** or the ***odd-even merge-sort***. The second name is derived from the fact that at each stage, the players in odd positions within the subgroups to be merged are separated from those in even positions.

**Example 7.1.3.** *Draw a chart for Dr. Ecco's solution to Problem 2 of section 1.2, analogous to the one given in Example 7.1.2.*

*Solution.*



□

Hackett's problem in Section 3.6 of *Ecco* bears a superficial resemblance to Coach McGraw's problem, in that both involve ranking in parallel processing.

There are however two significant differences. Hackett is trying to determine which of 10 different types of clamps is the lightest. He does not have to obtain a complete ranking, only to find the lightest type. Also, many copies of the same type are available. In this regard Hackett's problem is also reminiscent of the contaminated pill problem and Phoebe Fivewood's golfball problem in the exercises and questions at the end of the introduction to this notebook.

Again, however, there are significant differences: Hackett has several balances (not scales) at his disposal, and he is only allowed to weigh one clamp against another on each balance.

A two-way comparison requires one balance, a three-way comparison requires three balances (A *vs* B, A *vs* C, and B *vs* C), a four-way comparison requires that we compare every two of the four clamps, and so requires six balances, and so on. In sequential processing, there is no reason to use anything other than two-way comparisons. However, this is not the case in parallel processing, when the number of types still in contention shrinks with respect to the number of balances.

**Example 7.1.4.** *Suppose we only have 6 balances. What is the largest number of types of clamps from which we can identify the lightest type with three rounds of weighing?*

*Solution.* At the end of the third round, we must have only 1 type left. It follows that we can have at most 4 types left after the second round, so that in the third, we can do a four-way comparison. We may have as many as 9 types left after the first round. Then in the second, we can do one three-way comparison and three two-way comparisons. Hence there may be 15 types at the beginning. If we have 16 instead, then at least 10 will be left after the first round, 5 after the second and 2 after the third. □

We have another weighing problem in Section 5.8 of *Ecco*, but we use sequential processing here, and a scale instead of a balance. The interesting feature of this problem lies in the uncertainty of the exact weights of the coins.

**Example 7.1.5.** *If the total weight of four coins is 44 grams, we may have four real coins or three real and one fake. Find another ambiguous value of the total weight of our coins.*

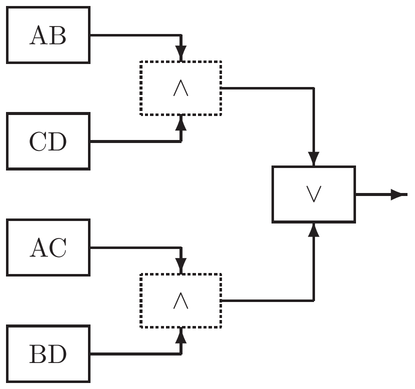
*Solution.* Suppose the total weight of four coins is 42.8 grams. We may have four fake coins each weighing 10.7 grams, or three fake each weighing 10.6, and one real weighing 11.0. □

## 7.2 Circuits, gates, and logic

We turn now to the problem of circuits in Section 3.4 of *Ecco*, using AND and OR gates. In the example of four signals, at most one of which may be "go", the circuit which checks all pairs uses 7 gates, and the maximum number of signals fed into a gate is 6. As was pointed out, this becomes terribly inefficient as the number of signals increases.

**Example 7.2.1.** *We have four signals, at most one of which may be "go". Design a circuit using 7 gates, with the maximum number of signals fed into a gate being 2.*

*Solution.* Let the signals be A, B, C and D. The following diagram shows such a circuit consisting of 5 OR gates, represented by solid boxes, and 2 AND gates, represented by dotted boxes.



This circuit is constructed as follows. The 4 signals are arranged in a 2 by 2 configuration as shown in the upper group. Signals on the same row are fed into the same OR gate. In the lower group, signals on the same column are fed into the same OR gate. It is easy to see that if at most one signal is "go", no alarm will be sounded. Suppose two signals are "go". They cannot simultaneously be on the same row and on the same column. It follows that these two signals are fed into different OR gates in at least one group. This will produce an output of 1 for the AND gate in that group, which is then fed through to the final OR gate to trigger the alarm. □

The problem is considerably more difficult if several signals may be "go". Suppose we use the idea in Example 6, where the circuits consists of groups of OR gates feeding into an AND gate, and the AND gates feeding into a final OR gate. Let $t$ be the minimum of "go" signals to trigger the alarm. Then we should have $t$ OR gates in each group.

Let $m$ be the total number of signals and $n$ be the number of groups. We can represent the circuit in the form of an $m \times n$ matrix, where the $(m, n)$-th entry of the matrix is one of 0, 1, $\ldots, t-1$, corresponding to the OR gate in the $n$-th group into which the $m$-th signal is fed.

Such a matrix has the following property. For any $t$ columns, there must exist at least one row which intersects them in $t$ distinct elements, meaning that these $t$ signals are fed into different OR gates. Conversely, a matrix with this property will yield a circuit that works.

For $t = 3$, there is the Denham matrix $D_n$ which has $n$ rows numbered $n, n-1, \ldots, 1$ from top to bottom and $2n$ columns numbered $-n, -(n-1), \ldots, -1, 1, 2, \ldots, n$ from left to right. We first construct the $n \times n$ submatrix on the right. For $1 \leq \ell \leq n$, the $(\ell, \ell)$-th entry is 1. For $1 \leq \ell < k \leq n$, the $(\ell, k)$-th entry is 0. For $1 \leq k < \ell \leq n$, the $(\ell, k)$-th entry is 0 if $k + \ell$ is odd and 2 if $k + \ell$ is even. The $n \times n$ submatrix on the left is obtained by reflecting the one on the right about their common border, and then interchanging 0's and 2's. The following diagram shows $D_8$, but $D_n$ for $1 \leq n \leq 7$ are all nested inside.

| | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Row 8 | 1 | 2 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 1 |
| Row 7 | 2 | 1 | 2 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 1 | 0 |
| Row 6 | 2 | 2 | 1 | 2 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 1 | 0 | 0 |
| Row 5 | 2 | 2 | 2 | 1 | 2 | 0 | 2 | 0 | 2 | 0 | 2 | 0 | 1 | 0 | 0 | 0 |
| Row 4 | 2 | 2 | 2 | 2 | 1 | 2 | 0 | 2 | 0 | 2 | 0 | 1 | 0 | 0 | 0 | 0 |
| Row 3 | 2 | 2 | 2 | 2 | 2 | 1 | 2 | 0 | 2 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| Row 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| Row 1 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Column | -8 | -7 | -6 | -5 | -4 | -3 | -2 | -1 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |

We now prove that for any three columns $i < j < k$ in $D_n, n \geq 2$, there exists a row $\ell$ such that the $(\ell, i), (\ell, j)$ and $(\ell, k)$-th entries are distinct. We use mathematical induction on $n$. The result is easily verified for $D_2$. Suppose it holds for $D_{n-1}$ for some $n \geq 3$. Consider $D_n$.

**Case 1:** $i > -n$ **and** $k < n$**.**
By the induction hypothesis, we may choose $\ell$ to be the value which works for $D_{n-1}$.

**Case 2:** $i = -n$ **and** $k = n$**.**
Take $\ell = |j|$.

**Case 3:** $i > -n$ **and** $k = n$**.**
If $ij > 0$ and $i + j$ is odd, or if $ij < 0$ and $i + j$ is even, take $\ell = n$. If $ij < 0$ and $i + j$ is odd, or if $ij > 0$ and $i + j$ is even, take $\ell = |j|$.

**Case 4:** $i = -n$ **and** $k < n$**.**
This is analogous to Case 3.

**Example 7.2.2.** *Solve the Omniheurists' Contest Problem #4 in Section 3.4.*

*Solution.* Decrypted, the statement of the problem reads: *Suppose that two "go" signals are permitted. But three or more are not. Draw a circuit that detects three or more using no more than 99 gates.* A 33-gate circuit may be constructed based on the Denhan matrix $D_8$. The 16 columns represent the signals. For $1 \leq \ell \leq n$, row $\ell$ represents a group consisting of 1 AND gate into which 3 OR gates feed. Signal $k$, where $-8 \leq k \leq -1$ or $1 \leq k \leq 8$, is fed into the 0-th, 1-st or 2-nd OR gate according to the the $(\ell, k)$-th entry in $D_n$. The 8 AND gates all feed into 1 final OR gate. □

The AND and OR gates are practical versions of ***Boolean functions*** whose domains are statements with ***truth values***, 1 for true and 0 for false. These two functions are symbolized as $\wedge$ and $\vee$ respectively. They are defined by the following ***truth tables***.

| $p$ | $q$ | $p \wedge q$ | $p \vee q$ |
|---|---|---|---|
| 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 |

In other words, $p \wedge q$ is true if and only if both $p$ and $q$ are true, and $p \vee q$ is false if and only if both $p$ and $q$ are false. Note that the OR function is inclusive, that is, it is true as long as either $p$ or $q$ is true, or both. There is also an exclusive form of the OR function in that the compound statement becomes false when both parts are true. This is in use in the problem in Section 2.2.

**Example 7.2.3.** *Analyze the problem in Section 2.2 of* Ecco *by means of a truth table.*

*Solution.* Let $A_1, A_2, A_3, B_1, B_2$ and $B_3$ denote the statements made by A and B respectively, that is

| | |
|---|---|
| $A_1$ : | Exactly one of W, X, and Y is true. |
| $A_2$ : | Exactly one of X, Y, and Z is true. |
| $A_3$ : | Exactly one of W and Z is false. |
| $B_1$ : | Exactly one of W, X, and Y is true. |
| $B_2$ : | Exactly one of X, Y, and Z is true. |
| $B_3$ : | Exactly one of W, Y, and Z is true. |

We have 16 combinations of truth values for $W, X, Y$ and $Z$. These are displayed in tabular form below, where 0 means "false" and 1 means "true":

| $W$ | $X$ | $Y$ | $Z$ | $A_1$ | $A_2$ | $A_3$ | $B_1$ | $B_2$ | $B_3$ |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 | 0 |
| 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | 1 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |
| 1 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |

Note that A's three statements are never true simultaneously in any row, so $A$ must be the liar. Also, there is exactly one row where all of B's answers are truthful, so B is the trustworthy spy and $Y$ is the only assertion which is true. □

The situation in Section 4.5 of *Ecco* is even simpler. The key observation is that if one unit claims that another one is faulty, then at least one of them must be faulty.

**Example 7.2.4.** *Suppose Dr. Bugunter had recorded the result incorrectly in that "good" and "faulty" were interchanged. At least how many units must be faulty?*

*Solution.* Suppose C is faulty. Then both E and A are also, since they claim that C is good. Then B must be faulty too since B claims that A is good, and the same reasoning shows that F is also faulty. Thus D is the only unit that may be good. Suppose C is good. Then so is E, and both D and F are faulty. Since B claims that C is faulty, B is in fact faulty. A may either be good or faulty. In conclusion, B, F and at least one other unit must be faulty. □

The problems in Sections 5.3, 3.5 and 4.1 of *Ecco* have a common solution, based on a structure known as the butterfly network.

**Example 7.2.5.** *Suppose in Section 5.3 of* Ecco, *there are four West Coast cities, each with mail for each of four East Coast cities. Two planes take off from each West Coast city. The planes met en route to exchange mail, each stopping at most once in between. At most two planes can land in any city, including any of the East Coast cities. Devise a plan to carry out the delivery.*

*Solution.* Let the East Coast cities be $A, B, C$ and $D$. In the first half, one plane from each West Coast city takes all mail for $A$ and $B$. These four meet in pairs, exchanging mail for $A$ and mail for $B$. The other plane from each West Coast city takes all mail for $C$ and $D$, meet in pairs and do a similar exchange. In the second half, the planes fly to their destinations, with two planes arriving at each East Coast city. □

**Example 7.2.6.** *Suppose in Section 3.5 of* Ecco *there are only four observation posts. Show how they can learn about all attacks in 2 minutes.*

*Solution.* Let the observation posts be $A, B, C$ and $D$. In the first minute, $A$ and $C$ exchange information while $B$ and $D$ do likewise. In the second minute, $A$ and $B$ exchange information while $C$ and $D$ do likewise. □

**Example 7.2.7.** *Suppose in Section 4.1 of* Ecco *there is a barrel of each of four types of toxic chemicals stored in each of four warehouses. The barrels must be sorted so that each warehouse contains four barrels of the same chemical. Show how to do this in 2 days using 2 trucks, with each warehouse participating in one exchange per day.*

*Solution.* Let the warehouses be $A, B, C$ and $D$, and the barrels of toxic chemicals be 1, 2, 3 and 4. During the first day, the trucks move barrels 3 and 4 from $A$ and $B$ to $C$ and $D$, bringing back barrels 1 and 2. For example, truck #1 picks up 3 and 4 from $A$, drops off 3 at C and picks up 1 from C, proceeds to D drops off 4 and picks up 1 from D. Truck #2 picks up barrels 3 and 4 from $B$, drops off 3 at C and picks up 2 from C, proceeds to D drops off 4 and picks up 2 from D. On the return trip that same day, the trucks drop the 1's at A and the 2's at B. During the second day, one truck exchanges barrels 1 and 2 between $A$ and $B$, while the other exchanges barrels 3 and 4 between $C$ and $D$. □

## 7.3 Exercises

1. In Section 1.4 of *Ecco*, get a complete ranking of 8 players in 16 hours using 1 court.
2. In Section 1.4 of *Ecco*, get a complete ranking of 7 players in 6 hours using 3 courts.
3. A tralance has three pans, and we may put one object on each pan. It will determine the lightest of the three. Suppose that in Section 3.6 of *Ecco* we can use 7 tralances for 2 rounds. What is the maximum number of types of clamps of different weights such that we can determine the lightest type among them?
4. There are 10 coins in each of 10 stacks. Nine of the stacks contain only real coins, each of which weighs 11 gram. The remaining stack contains only fake coins, each of which weighs 10.9 grams. We have a scale which can determine the total weight of any number of coins in one weighing. What is the minimum number of times we have to use the scale to find out which stack contains fake coins?

5. In Section 3.4 of *Ecco*, if there are only 4 signals, at most 1 of which can be "go", design a checking circuit using only 6 gates, with at most 3 signals fed into any gate.

6. Design a checking circuit using 13 gates, if there are 6 signals and at most 2 of them can be "go".

7. In Section 2.2 of *Ecco*, suppose that spies A and B make the following statements instead.

| | |
|---|---|
| A reports: | Exactly two of W, X and Y are true. |
| | Exactly two of X, Y and Z are true. |
| | Exactly one of W and Z is false. |
| B reports: | Exactly two of W, X and Y are true. |
| | Exactly two of X, Y and Z are true. |
| | Exactly two of W, Y and Z are true. |

   Determine which spy tells only the truth, and which assertions are true.

8. In Section 4.5 of *Ecco*, suppose that A says that C is faulty; B says that either A is faulty or C is good; C says that either D is good or E is faulty; D says that F is good; E says that either C is faulty or F is good; and F says that B is faulty. What is the maximum number of units that are good? Which ones are they?

9. In Section 5.3 of *Ecco*, suppose that there are three West Coast cities, each with mail for each of three East Coast cities. Two planes take off from each West Coast city. The planes met en route to exchange mail, each stopping at most once in between. At most two planes can land in any city, including the East Coast cities. Devise a plan to carry out the delivery.

10. In Section 3.5 of *Ecco*, it takes longer for 3 observation posts to learn about all attacks than for 4 observation posts. What is the next smallest example of this anomaly?

11. In Section 4.1 of *Ecco*, suppose that there is a barrel of each of five types of toxic chemicals stored in each of five warehouses. Using two trucks, what is the minimum number of days needed to sort the barrels so that each warehouse contains five barrels of the same chemical if each warehouse may only participate in one exchange per day?

## 7.4 Questions

1. In Section 1.4 of *Ecco*, get a complete ranking of 5 players in 5 hours using 2 courts, assuming that only one court is available for the fifth hour.
2. Solve Coach McGraw's problem in 9 hours using 2 courts.
3. In Section 3.6 of *Ecco*, suppose we have 22 types of clamps and 10 balances. What is the minimum time required to determine the lightest type? Show how, and explain why it cannot be less.
4. Given 4 coins, Dr. Ecco's method in Section 5.8 determines the nature of each coin in at most 3 weighings. If we know in advance the number $n$ of fakes among the 4, what is the minimum number of weighings required for each value of $n$? Show how, and explain why it cannot be less.
5. In Section 3.4 of *Ecco*, if there are only 4 signals, at most 1 of which can be "go", design a checking circuit using only 5 gates.
6. Solve Lars Pollard's problem using 17 gates, if there are 9 signals, at most 2 of which can be "go".
7. In Section 4.5 of *Ecco*, suppose exactly three of Dr. Bugunter's units are faulty. Which ones can they be? Find all solutions and prove that there are no others.
8. Suppose in Section 2.2, spies A and B make the following statements instead.

| | |
|---|---|
| A reports: | Exactly one of W, X and Y is false. |
| | Exactly one of X, Y and Z is false. |
| | Exactly one of W and Z is true. |
| B reports: | Exactly one of W, X and Y is false. |
| | Exactly one of X, Y and Z is false. |
| | Exactly one of W, Y and Z is false. |

   Determine which spy tells only the truth, and which assertions are true.

9. Solve General Evans' problem in Section 5.3 of *Ecco*, but with 6 cities on each coast and 12 planes, so that the largest number of intermediate stops made by any plane is as small as possible.
10. In Section 3.5 of *Ecco*, suppose there are 6 observation posts. What is the minimum number of calls, not rounds, needed for them to learn about all attacks?
11. Solve Mr. Barin's problem in Section 4.1 of *Ecco* using 3 trucks, if there are 6 warehouses and 6 kinds of toxic chemicals. What is the minimum number of days required?
12. Suppose there are 9 warehouses and 9 kinds of toxic chemicals. Each warehouse has 1 barrel of all 9, and we want each to have all 9 barrels of a kind. We may hire 9 trucks. On any day, each may make a round trip between 2 warehouses, at most 2 can visit any one warehouse, and each barrel may only be moved once. Show how this can be done in 2 days.

# A Appendix

## A.1 The pigeonhole principle

*This section is an adaptation of the discussion on the pigeonhole principle from* The Hungarian problem book *by Anton Cherney and Andy Liu.*

The finiteness of a set gives rise to many of its basic properties. We present some of the most important results. Let us assume that all sets we deal with here are *non-empty*, unless explicitly stated otherwise.

An important property of finite sets of real numbers is the ***extremal value principle***. It states that every finite set of real numbers has a maximum and a minimum. We can give a proof of the existence of a minimum as follows. Pick any two of the numbers and throw away the larger one, or either one if they are equal. Repeat this process. Since the set of numbers is *finite*, the process eventually terminates, and the number we still have is a minimum.

Given finite set of real numbers, $S = \{x_1, x_2, \ldots, x_n\}$, the average, or better, the ***arithmetic mean*** of $S$ is the number $M_1$ defined by:

$$M_1(S) = \frac{x_1 + x_2 + \cdots + x_n}{n}.$$

The ***power means inequality*** says that

$$\min(S) \leq M_1(S) \leq \max(S)\,,$$

where $\min(S)$ and $\max(S)$ denote the smallest (minimum) and largest (maximum) numbers in $S$ respectively.

The proof of the inequality is fairly straight forward. If, for example, $x$ is the minimum of $S$, then $x \leq x_i$ for all $x_i$, so

$$nx \leq x_1 + x_2 + \cdots + x_n$$

and dividing by $n$, we have

$$\min(S) = \frac{nx}{n} \leq \frac{x_1 + x_2 + \cdots + x_n}{n} = M_1(S)$$

and the proof that $M_1(S) \leq \max(S)$ is similar.

The ***mean value principle*** states that in every finite set of real numbers, there is at least one which is not less than the arithmetic mean of the set, and at least one not greater. This follows from the Extremal Value Principle and the Power Means Inequality, since a maximum and a minimum of the set can play the roles of these two numbers.

A most important corollary of the Mean Value Principle is the **pigeonhole principle**. The simple version and the extended version stated in Chapter 1 can be combined as follows:

> If a finite number of pigeons enter a finite number of pigeonholes, and if there are more pigeons than pigeonholes, then the average number of pigeons per pigeonhole is greater than one and at least one pigeonhole contains no less than the average.

The Extremal Value Principle and its corollaries do not hold if we remove the assumption that the set of numbers we are dealing with is finite. If we restrict our attention to the set of nonnegative integers, we can salvage a partial result. The **well-ordering principle** states that every nonempty set of nonnegative integers has a minimum.

## A.2 Multiples, divisors, and congruences

This section establishes the basic facts about multiples, divisors, and relatively prime numbers mentioned in Chapter 1, and you may refer to page 12 for the terminology.

Certain facts about divisibility are fairly obvious, and not very difficult to prove:

**Theorem A.2.1 (Basic properties of divisibility).**

1. *If $d \mid b$ and $b \mid c$ then $d \mid c$. (That is,* ***divisibility is transitive.****)*

2. *If $d \mid b$ then $d \mid nb$ for any integer $n$.*

3. *If $a = b + c$, and if $d$ divides any two of $a$, $b$, and $c$, then $d$ also divides the third one.*

One fact that is not so easy to prove is the following:

**Theorem A.2.2 (The Linear Divisibility Theorem).** *Given positive integers $a$ and $b$, the smallest positive value of $ax + by$ for all integers $x$ and $y$ is the greatest common divisor of $a$ and $b$.*

For example, the smallest positive value of $15x + 24y$ is 3, which occurs when $x = -3$ and $y = 2$ (or when $x = 5$ and $y = -3$, or for many other values), and 3 happens to be the greatest common divisor of 15 and 24.

The proof is below — It is not constructive, which means that it does not tell you how find $x$ and $y$. That would be a useful thing to be able to do. The algorithm that enables us to find $x$ and $y$ is often called the ***extended Euclidean algorithm*** and it is dealt with in Chapter 3 (see page 37). At this point, all we need to know is that there is an $x$ and $y$. Here is a proof of the Linear Divisibility Theorem.

*Proof.* Let $k$ be the smallest positive value of $ax + by$ (the well-ordering principle guarantees that $k$ exists). We will prove the theorem in two steps. First, we will show that $k$ is at least as small as both $a$ and $b$. Second we will show that if $k > d$, where $d$ is the greatest common divisor of $a$ and $b$, then we can find a positive value for $ax + by$ that is smaller than $k$ which contradicts the definition of $k$.

*Step 1.* $k \le a$ and $k \le b$: This is clear because with $x = 1$ and $y = 0$ we get $ax + by = a$, so $k \le a$. Similarly, with $x = 0$ and $y = 1$ we get $ax + by = b$, so $k \le b$.

*Step 2.* If $k > d$, then we can find $x$ and $y$ so that $ax + by < k$:

To see why, let us suppose that $x'$ and $y'$ are integers for which

$$ax' + by' = k. \tag{A.1}$$

Now, $k$ must be strictly less that the smaller of $a$ and $b$. Moreover, $k$ cannot divide both $a$ and $b$ (because our assumption is that $k$ is larger than the greatest common divisor of $a$

and $b$). So let us suppose that $k$ does not divide $b$. This means that $k \neq b$ and so by step 1, we have $k < b$. Then by the division algorithm,

$$b - kq = r, \qquad 0 < r < k. \tag{A.2}$$

Substitute the value of $k$ from equation (A.1) into equation (A.2) to get:

$$b - (ax' + by')q = r$$

This implies that $a(-x'q) + b(1 + y'q) = r$, which shows that if $x = -x'q$ and $y = 1 - y'$, we get a value of $ax + by$ that is positive but strictly smaller than $k$.

This finishes step 2 and completes the proof. □

### A.2.1 Two important consequences.

In general, if $d_1$ and $d_2$ are common divisors of $a$ and $b$, there is not much that we can conclude about the relationship between $d_1$ and $d_2$. However if one of them is the greatest common divisor of $a$ and $b$, then it must be a multiple of the other. As an illustration, consider the numbers 60 and 72. Their greatest common divisor is 12, the other common divisors are 1, 2, 3, 4, 6, and 12; and all of them divide 12.

The proof that this always happens follows very quickly from the Linear Divisibility Theorem:

**Theorem A.2.3 (The Common Divisor Theorem).** *If $d$ is a common divisor of $a$ and $b$, then $d \mid \gcd(a, b)$. The converse is also true.*

*Proof.* By Theorem A.2.2, there is some value of $x$ and $y$ such that $ax + by = \gcd(a, b)$. By the basic properties of divisibility every divisor of $a$ and $b$ also divides $ax + by$. □

The second consequence is the one the gets to the heart of the matter about the cancellation law and congruences.

**Theorem A.2.4 (The Relatively Prime Divisor Theorem).** *Suppose $\gcd(d, a) = 1$ and that $d \mid ab$. Then $d|b$.*

*Proof.* By Theorem A.2.2, there are integers $x$ and $y$ such that $dx + ay = 1$. If we multiply the equation by $b$ we get $dbx + aby = b$. Now $d \mid dx$ and $d \mid aby$, so by the basic properties of divisibility, $d$ must also divide $b$. □

**Example A.2.5.** *Show that if $4 \mid c$ and $9 \mid c$, then $36|c$.*

*Solution.* Since $4 \mid c$, we have $c = 4a$ for some integer $a$. This means that $9 \mid 4a$, and since $\gcd(4, 9) = 1$ we can conclude that $9 \mid a$. Then $a = 9b$ for some integer $b$, so

$$c = 4a = 4(9b) = 36b,$$

showing that $36 \mid c$. □

***Why the cancellation law works***

Let us examine what happens for the congruence (1.1) in Chapter 1, that is for

$$4a \equiv 4b \pmod{9}.$$

The congruence means that $a$ and $b$ are related by the *equation*

$$4a = 4b + 9m\,. \tag{A.3}$$

Since the left hand side is a number that is divisible by 4, then the right hand side must also be divisible by 4. That is, $4 \mid (4b - 9m)$. Since $4 \mid 4b$, we must conclude that $4 \mid 9m$, and it follows from the Relatively Prime Divisor Theorem that $4 \mid m$. Consequently, $m = 4k$ for some integer $k$ and substituting this into equation (A.3), we get

$$4a = 4b + 9(4k)$$

and so $a = b + 9k$, or equivalently, $a \equiv b \pmod{9}$.

Now try the same thing with the congruence (1.2). That congruence means that

$$4a = 4b + 6m\,, \tag{A.4}$$

As before, the right hand side must be divisible by 4, and so $4 \mid 6m$. But since 4 and 6 are not relatively prime, we cannot conclude that 4 divides $m$. In other words, it may not be true that $6m = 6(4k)$, and so we cannot conclude that $a \equiv b \pmod{6}$.

## A.3 We've got your number

Your life is filled with code numbers. Every commercial product has a 12-digit number called the UPC (the Universal Product Code). The UPC number is written as a barcode so it can be read by scanners, and in decimal form so it can be read by humans. Soon, all products will carry a 13-digit barcode number that looks a lot like a UPC—in fact, it is a superset of the UPC called the EAN (European Article Number). To order a book, you may have to supply its ISBN (International Standard Book Number). To subscribe to a magazine, you may be asked for its ISSN (International Standard Serial Number).

Open your wallet and check your student ID card. It likely has a code number on it. Your driver's permit has a 'licence number' and it is probably accompanied by a barcode or a magnetic strip as well.

Your credit card has a 16 digit code on it. If you order something over the internet, you will be asked to provide that code. If you make a mistake entering the digits, you will see a message like, "*Invalid VISA number! Please check the number and re-enter.*"

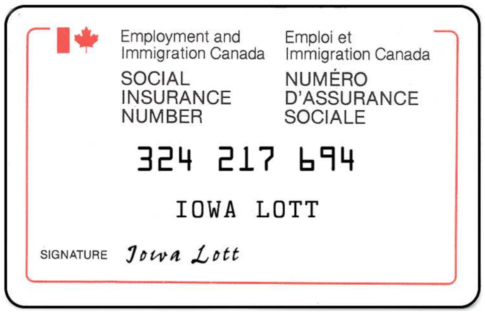
A few years ago, I encountered a similar situation when I was using a computer program to prepare my income tax return. I wanted to see what the tax would be for variety of taxpayers, so I made up some ficticious data for "Richard Richman." As part of the data, I included my own Social Insurance Number. The program promptly rejected this because my S.I.N. was already in the small database it was creating on my hard drive, and two people cannot have the same number. So, I made up a completely arbitrary one, but that didn't get me very far—the program told me that I had entered an invalid number, and it wouldn't let me continue until I provided an acceptable one.

How can an on-line book company tell when you have entered a incorrect VISA number? How did the income tax program know that I was entering an fake S.I.N? This "magic" is accomplished by using what is called an *error detecting code* and, as is true of all magic, the idea behind error detection is quite simple.[1]

[1] When you listen to a CD, the music is brought to you courtesy of an *error correcting code*. That code not only detects errors, it also repairs them. The digitally encoded music on a CD has such strong error correcting capabilities that apparently you can drill a 2.5 mm hole through your CD and it will still play flawlessly. Prof. Scarlet does **not** advocate that you try this!

## The IBM scheme

For validation, most error-detecting schemes use a *check-digit*. This is usually the rightmost digit of the code. The other digits, the *information digits*, can be freely chosen, but the check digit is calculated. For Canadian S.I.N.'s and for many credit cards, the check-digit is computed using a method devised by IBM. Spaces have no significance and are only there to make it easier to read the number.


Employment and Immigration Canada
Emploi et Immigration Canada
SOCIAL INSURANCE NUMBER
NUMÉRO D'ASSURANCE SOCIALE
324 217 694
IOWA LOTT
SIGNATURE Iowa Lott

Here is how the IBM scheme is used to validate Iowa Lott's S.I.N. Beginning with the rightmost check-digit, identify the alternate digits. I have put them in boxes.

$$\boxed{3}\,2\,\boxed{4}\,2\,\boxed{1}\,7\,\boxed{6}\,9\,\boxed{4}$$

Add the boxed digits: $3 + 4 + 1 + 6 + 4 = 18$

Multiply the other digits by 2: 4, 4, 14, 18

Add the *digits* of these numbers:

$$4 + 4 + 1 + 4 + 1 + 8 = 22$$

Add the two results: $18 + 22 = 40.$

(Note that in the third step we do not add the numbers, rather we add the *digits* of the numbers.) The S.I.N. is considered to be valid if the result is divisible by 10, and so Iowa Lott's number passes the validation test.

## Calculating the check-digit

The validation procedure tells us how the check digit is found: carry out the calculations with $x$ in the place of the check digit and solve for $x$. For example, suppose the information digits for your S.I.N. are 22501008. Then your S.I.N. will be 225-010-08$x$, and $x$ is calculated as follows:

Identify the alternate digits:

$$\boxed{2}\,2\,\boxed{5}\,0\,\boxed{1}\,0\,\boxed{0}\,8\,\boxed{x}$$

| | |
|---|---|
| Add the boxed digits: | $2+5+1+0+x=8+x$ |
| Multiply the other digits by 2: | $4,\ 0,\ 0,\ 16$ |
| Add the *digits* of these numbers: | $4+1+6=11$ |
| Add the two results: | $=19+x$ |

To make $19+x$ divisible by 10, the digit $x$ must be 1, and the social insurance number would become 225-010-081.

## How good is the error detection?

The most common errors in entering numbers are reported to be:

- entering one of the digits incorrectly; or
- interchanging two adjacent digits.

No error detection scheme can flag all errors, and so they are designed to catch only the most common ones. At the very least, an error detection scheme should flag either of the above.

The IBM method will detect an error if a single digit is changed. This includes the case where the check digit is changed.

To illustrate why, let us see what happens if the digit 7 on Iowa Lott's credit card is changed to something else (see picture on page 126). That is, suppose that instead of entering 7, you enter an $x$, and this is the only error that you make.

The credit card number is

$$4\,\boxed{0}\,0\,\boxed{2}\,1\,\boxed{2}\,6\,\boxed{5}\,x\,\boxed{0}\,2\,\boxed{1}\,0\,\boxed{6}\,9\,\boxed{3}\,.$$

The position of the digit $x$ means that it is one of the digits that will be multiplied by 2 during the validation process. Depending upon $x$, the number $2x$ could be either a single digit or a double digit number. We consider each case separately.

If $0 \leq x < 5$ (so $2x$ is a single digit).
Carrying out the IBM validation procedure, the final sum will be $45+2x$ (try it). No matter what the digit $x$ is, this will not be divisible by 10, and so an error will be detected.

If $5 \leq x \leq 9$ (so $2x$ is a two-digit number).
The digits of $2x$ will be 1 and $2x - 10$. Carry out the validation and you will get a final sum of $36 + 2x$. Since $x \neq 7$ and since $5 \leq x \leq 9$, this sum will also fail to be divisible by 10, and an error will be detected.

The IBM method is very good at detecting an error if a single digit is entered incorrectly. Although it is not completely successful in detecting interchanges of adjacent digits, it will detect most of those errors as well.

The IBM method will detect an error if two adjacent digits are interchanged, *provided the two digits are not 9 and 0.*

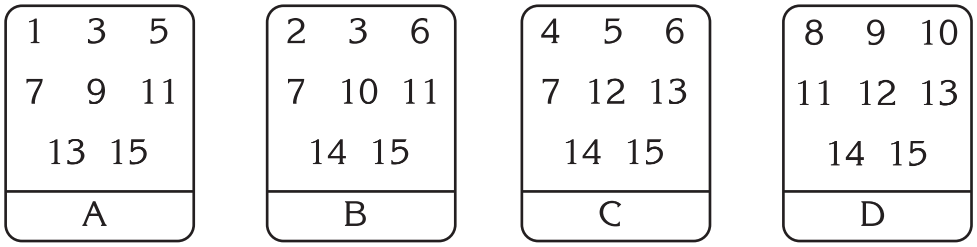
We leave it to the reader to verify the statement.

The IBM method is not the only one that is used for error detection. UPC and EAN numbers use a similar scheme. (For both codes, alternate numbers are tripled instead of doubled, and the numbers, not the digits, are added. Numbers are considered valid if they are divisible by 10). The ISBN error detection method is based on divisibility by 11. It will detect an error if a single digit is changed, or if two digits are swapped, even if the digits are not adjacent.

Error correction requires more check-digits than error detection. On page 126, it was mentioned that CD's have very strong error correction capabilities. This comes at a price — most of the binary digits on a CD are check-digits. Only about one-third of the data is music. The stronger the error correction, the more redundancy (check-digits) is required. Conversely, if there is lot of redundancy in the data, that is a signal there might be error correction going on. Do you not find it interesting that geneticists working on the human genome project have been quoted as saying there seems to be a lot of extra junk and redundancy in our DNA? In the real world, mathematics can be found everywhere.

## A.4 Binary Numbers – what are they?

### A simple trick



A spectator writes a number from one to fifteen on a piece of paper and shows it to the audience but not to you. You then ask the spectator to name each of the cards on which the number appears. For example, if the secret number is 11, the spectator will say that the number is on cards “A”, “B”, and “D”. Then you immediately tell the audience what the secret number is. This trick may be repeated several times.

***How it is done***

You add the numbers in the upper left hand corners of the named cards. For example the numbers on cards ”A”, ”B”, and ”D” are 1, 2, and 8, so the secret number is 11.

### Binary Numbers

Knowing how it is done doesn't explain what numbers should be on what cards. The distribution of the numbers is determined by using their binary expansions.

Every number from 1 to 15 can be written as a sum of the numbers 1, 2, 4, and 8. In fact every positive integer can be written *in a unique way* as the sum of powers of 2.

The following example is not a proof, but it should convince you that the preceding statement is true:

**Example A.4.1.** *Express 221 as the sums of powers of two.*

To solve this, first write powers of 2:

$$1 \qquad 2 \qquad 4 \qquad 8 \qquad 16 \qquad 32 \qquad 64 \qquad 128 \qquad 256\ldots$$

We stop here because $128 < 221 < 256$, that is, because 128 is the largest power of two that does not exceed 221. This means that 128 is one of the powers of two that will be used to create the sum. We remove 128 from 221, and then look for the next largest power that can be removed. The full process looks like this:

$$\begin{aligned}221 &= 128 + 93\\ &= 128 + 64 + 29\\ &= 128 + 64 + 16 + 13\\ &\vdots\\ &= 128 + 64 + 16 + 8 + 4 + 1\end{aligned}$$

or, writing it as actual powers:

$$231 = 2^7 + 2^6 + 2^4 + 2^3 + +2^2 + 2^0$$

If we are going to write lots of numbers as powers of 2, it is tedious to use the preceding notation. Instead, we use a positional notation to indicate which powers of two are present, the rightmost position being for $2^0$. We put 1's or 0's in the position according to whether the indicated power of 2 appears or does not appear in the expansion obtained. So, for 221, you would get

$$\begin{array}{cccccccc}\boxed{1} & \boxed{1} & \boxed{0} & \boxed{1} & \boxed{1} & \boxed{1} & \boxed{0} & \boxed{1}\\ 2^7 & 2^6 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0\end{array}$$

So, instead of writing $2^7 + 2^6 + 2^4 + 2^3 + +2^2 + 2^0$, we could simply write 11011101, which is called the *binary expansion* or *binary representation*, or *base-2* representation of 221. When different bases are involved, it is a common practice to indicate the base by using a parenthetical subscript, so we would write

$$221_{(10)} = 11011101_{(2)}\ .$$

The same idea is used to express numbers to other bases.

**Example A.4.2.** *Express 221 as a base 3 number.*

Solution

The powers of 3 are

$$\begin{array}{cccccccc}1 & 3 & 9 & 27 & 81 & 243 & \ldots\\ 3^0 & 3^1 & 3^2 & 3^3 & 3^4 & 3^5 &\end{array}$$

Proceed as before, this time using the division algorithm to obtain multiples of powers of three.

$$\begin{aligned}221 &= 2(81) + 59\\ &= 2(81) + 2(27) + 5\\ &\vdots\\ &= 2(81) + 2(27) + 1(3) + 2(1)\\ &= 2(3^4) + 2(3^3) + 1(3^1) + 2(3^0),\end{aligned}$$

and so

$$221_{(10)} = 22012_{(3)}$$

If $k$ is a positive integer bigger than 1, we use the same process to convert a base 10 number to base $k$. Note that base-$k$ integers use the digits $0, 1, 2, \ldots, (k-1)$, so

base-2 uses digits 0 and 1
base-3 uses digits 0, 1 and 2
base-10 uses digits $0, 1, 2, \ldots, 8, 9$
base-16 uses digits $0, 1, 2, \ldots, 8, 9, A, B, C, D, E, F$

## Counting in different bases

Assuming you are familiar with an odometer, you know how to count in base-$k$ — when the rightmost digit (the units digit) becomes $(k-1)$, a "rollover" occurs as in the following table.

| base 10 | base 2 | base 3 |
|---:|---:|---:|
| 0 | 0 | 0 |
| 1 | 1 | 1 |
| 2 | 10 | 2 |
| 3 | 11 | 10 |
| 4 | 100 | 11 |
| 5 | 101 | 12 |
| 6 | 110 | 20 |
| 7 | 111 | 21 |
| 8 | 1000 | 22 |
| 9 | 1001 | 100 |
| 10 | 1010 | 101 |
| 11 | 1011 | 102 |
| 12 | 1100 | 110 |
| 13 | 1101 | 111 |
| 14 | 1110 | 112 |
| 15 | 1111 | 120 |

**Example A.4.3.** *Is the number* $334212_{(5)}$ *odd or even?*

Being odd or even does not depend upon the base. A number is defined to be *even* if it is divisible by 2, otherwise it is defined to be *odd*.

Expanding the number $334212_{(5)}$ to base-10, we get

$$\begin{aligned} 334212_{(5)} &= 3(5^5) + 3(5^4) + 4(5^3) + 2(5^2) + 1(5) + 2 \\ &= \text{odd} + \text{odd} + \text{even} + \text{even} + \text{odd} + \text{even}. \end{aligned}$$

So $334212_{(5)}$ is an odd number.

## Explanation of the trick

The following table shows the base-2 representation of the numbers from 1 to 15. On card A we put the numbers whose binary representation has a 1 in the column labelled by A. On card B we put the numbers whose binary representation has a 1 in the column labelled by B. We do the same for C and D.

| base 10 | base 2 | | | |
|---|---|---|---|---|
| 1 | 0 | 0 | 0 | 1 |
| 2 | 0 | 0 | 1 | 0 |
| 3 | 0 | 0 | 1 | 1 |
| 4 | 0 | 1 | 0 | 0 |
| 5 | 0 | 1 | 0 | 1 |
| 6 | 0 | 1 | 1 | 0 |
| 7 | 0 | 1 | 1 | 1 |
| 8 | 1 | 0 | 0 | 0 |
| 9 | 1 | 0 | 0 | 1 |
| 10 | 1 | 0 | 1 | 0 |
| 11 | 1 | 0 | 1 | 1 |
| 12 | 1 | 1 | 0 | 0 |
| 13 | 1 | 1 | 0 | 1 |
| 14 | 1 | 1 | 1 | 0 |
| 15 | 1 | 1 | 1 | 1 |
| | D | C | B | A |

So, when I say if you tell me what cards your number is on, then I can tell you what your number is, I am really saying that if you tell me what your number is in base-2, then I can tell you what it is in base-10.

# B Solutions for the exercises

## B.1 Introduction

1. The chosen card will be the middle (4-th from the end) card in the column identified. The reason is that when you first pick up the cards, before you deal them the second time the chosen card is one of positions 8 through 14. When you deal these, the columns look like the following where the chosen card is one of #8 through #14 and where the o's indicate other cards.

| | | |
|---|---|---|
| o | o | o |
| o | o | o |
| o | #8 | #9 |
| #10 | #11 | #12 |
| #13 | #14 | o |
| o | o | o |
| o | o | o |

   Then when the cards are picked up the second time, the chosen card is now in one of the following positions:

   position 11 or 12 if it was in the first column

   position 10, 11, or 12 if it is in the second column, or

   position 10 or 11 if it was in the third column.

   When the cards are dealt for the last time the cards in positions 10, 11, and 12 become the middle cards as shown in the table.

2. Let the 10 colours be $C_1$ through $C_{10}$. On the scale place one pill of colour $C_1$, 2 pills of colour $C_2$, 3 pills of colour $C_3$ and so on. Record the total weight. It ends in $.000n$, and the pills of colour $C_n$ are the contaminated ones.

3. I'll do it for a four digit number — the proof for the other cases will be the same. Let the number be $abcd$, where $a$, $b$, $c$ and $d$ are the digits. The sum of the digits is $a+b+c+d$ and the number is actually $1000a+100b+10c+d$, so we want to show that
$$1000a+100b+10c+d \equiv a+b+c+d \pmod 3.$$

   In modular three arithmetic we have
$$1000 \equiv 1, \qquad 100 \equiv 1, \qquad 10 \equiv 1, \qquad \text{and } 1 \equiv 1 \pmod 3.$$

   Multiplying these congruences by $a$, $b$, $c$, and $d$ respectively we get:
$$1000a \equiv a, \qquad 100b \equiv b, \qquad 10c \equiv c, \qquad \text{and } d \equiv d \pmod 3.$$

Adding the four together we get

$$1000a + 100b + 10c + d \equiv a + b + c + d \pmod{3}.$$

4. Suppose the number is $fedcba$, that is, $100000f + 10000e + 1000d + 100c + 10b + a$. We want to show that

$$100000f + 10000e + 1000d + 100c + 10b + a \equiv a - b + c - d + e - f \pmod{11}.$$

In modular eleven arithmetic we have

$$\begin{aligned} 1 &\equiv 1 \\ 10 &\equiv -1 \\ 100 &\equiv 1 \\ 1000 &\equiv -1 \\ 10000 &\equiv 1 \\ 100000 &\equiv -1 \end{aligned}$$

where all congruences are mod 11.

Multiplying the congruences by $a$ through $f$ respectively, and adding them all together we get the desired result.

## B.2 Coding Theory

1. The table of Hamming distances is as follows:

| | 0000 | 0011 | 0101 | 0110 | 1001 | 1010 | 1100 | 1111 |
|---|---|---|---|---|---|---|---|---|
| 0000 | 0 | 2 | 2 | 2 | 2 | 2 | 2 | 4 |
| 0011 | 2 | 0 | 2 | 2 | 2 | 2 | 4 | 2 |
| 0101 | 2 | 2 | 0 | 2 | 2 | 4 | 2 | 2 |
| 0110 | 2 | 2 | 2 | 0 | 4 | 2 | 2 | 2 |
| 1001 | 2 | 2 | 2 | 4 | 0 | 2 | 2 | 2 |
| 1010 | 2 | 2 | 4 | 2 | 2 | 0 | 2 | 2 |
| 1100 | 2 | 4 | 2 | 2 | 2 | 2 | 0 | 2 |
| 1111 | 4 | 2 | 2 | 2 | 2 | 2 | 2 | 0 |

2.

$$\begin{array}{ccccccccccc|cccc}
a & a & a & a & & a & a & a & & & & a & & & \\
b & b & b & & b & b & & & b & b & & & b & & \\
c & c & & c & c & & c & & c & & c & & & c & \\
d & & d & d & d & & & d & & d & d & & & & d \\
\hline
1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0
\end{array}$$

Checking the total numbers of 1's under the columns containing $a$. $b$, $c$ and $d$ respectively, we obtain 4, 5, 5 and 2. Hence an error has occurred, at the digit corresponding to the subset $\{b, c\}$. It follows that the correct eleven-digit message is 11000010010.

3. (a) We have $10 \cdot 0 + 9 \cdot 7 + 8 \cdot 1 + 7 \cdot 6 + 6 \cdot 7 + 5 \cdot 2 + 4 \cdot 3 + 3 \cdot 1 + 2 \cdot 4 = 188 \equiv 1 \pmod{11}$. Hence the tenth digit is $X$.

   (b) We illustrate what happens if digit $e$ in *abcdefghi* is changed to $e'$ and if the check digit $j$ remains the same. We then have

$$10a + 9b + 8c + 7d + 6e + 5f + 4g + 3h + 2i + j \equiv 0 \pmod{11}, \qquad \text{and}$$
$$10a + 9b + 8c + 7d + 6e' + 5f + 4g + 3h + 2i + j \equiv 0 \pmod{11}.$$

   Subtracting the congruences, we get $6(e - e') \equiv 0 \pmod{11}$.
   Since 6 and 11 are relatively prime, we can divide by 6 and get $e - e' \equiv 0 \pmod{11}$, and since both $e$ and $e'$ are 'integers' from 0 to $X$, we must have $e = e'$. So the only way the number passes the test is if $e = e'$. The same reasoning holds for all other digits.

   (c) Yes. Suppose that $x$ and $y$ are adjacent digits that are switched and that the number passes the test in both cases, that is,

$$10a + \cdots + kx + (k-1)y + \cdots + j \equiv 0 \pmod{11}, \qquad \text{and}$$
$$10a + \cdots + ky + (k-1)x + \cdots + j \equiv 0 \pmod{11}.$$

   Subtracting, we get $k(x - y) + (k - 1)(y - x) \equiv 0 \pmod{11}$, that is, $x - y \equiv 0 \pmod{11}$. This can only happen if $x = y$.

   (d) No. For example the numbers 1001001001 and 1001002008 are pass the test for being ISBN numbers. Suppose now you receive the number 1001002001. This number is within Hamming distance 1 of both ISBN numbers, and so there is no way to tell which of 1001001001 or 1001002008 is the correct number.

## B.3 Cryptography

1. This short story does not contain a single appearance of the letter $e$.

2. Knowing that we have a generalized Caesar's Code, we check the first part of the message as follows:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| A | L | D | C | L | O | P | X | I | B |
| B | M | E | D | M | P | Q | Y | J | C |
| C | N | F | E | N | Q | R | Z | K | D |
| D | O | G | F | O | R | S | A | L | E |

   We can stop here as we come up with something that makes sense. The complete message is *Dog for Sale. Eats anything and is fond of children.*

3. Since the inverse of 7 is 15 in arithmetic modulo 26, the decoding function is $D(x) = 15(x - 1)$. We construct the decoding table as follows:

| A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| 1 | 8 | 15 | 22 | 3 | 10 | 17 | 24 | 5 | 12 | 19 | 0 | 7 |
| b | i | p | w | d | k | r | y | f | m | t | a | h |
| **N** | **O** | **P** | **Q** | **R** | **S** | **T** | **U** | **V** | **W** | **X** | **Y** | **Z** |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |
| 14 | 21 | 2 | 9 | 16 | 23 | 4 | 11 | 18 | 25 | 6 | 13 | 20 |
| o | v | c | j | q | x | e | l | s | z | g | n | u |

The decrypted message is *Semi-Annual After-Christmas Sale.*

4. Since we just know that the code is mono-alphabetic, we use statistical analysis as an aide. The frequency table is as follows:

| A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 28 | 37 | 11 | 4 | 10 | 2 | 11 | 1 | 30 | 7 | 10 | 7 | 59 |
| **N** | **O** | **P** | **Q** | **R** | **S** | **T** | **U** | **V** | **W** | **X** | **Y** | **Z** |
| 16 | 11 | 20 | 37 | 0 | 2 | 15 | 7 | 33 | 47 | 11 | 1 | 40 |

It would appear that M stands for $e$. This is reinforced by the frequent appearance of the three-letter word BPM, likely to be *the*. Pursuing this line of attack, we obtain the following decrypted message. *Sailing is the fastest growing participation sport of modern times. It is no longer the exclusive preserve of the very rich, nor does one has to be wealthy to own one's own craft. Sailing is a family sport. There is no room for the generation gap in a boat. Whether your preference is for the feel of a capricious breeze in your canvas, or for the heady excitement of a surging power boat, whether you find yourself drawn irresistably to the open sea, or take your pleasure in inland waters, there is nothing quite like the thrill of being in sole command of your own boat.* (Sometimes coders commit spelling errors.)

5. (a) Since $\gcd(637, 14) = 7$, the integers 637 and 14 are not relatively prime so there is no inverse function.

   (b) Using the Euclidean Algorithm to find $\gcd(126, 55)$ gives:

$$\begin{aligned} 126 &= 2 \cdot 55 + 16, \\ 55 &= 3 \cdot 16 + 7, \\ 16 &= 2 \cdot 7 + 2, \\ 7 &= 3 \cdot 2 + 1 \end{aligned}$$

   so $\gcd(126, 55) = 1$, and there is an inverse.
   Backstepping through the algorithm, we get

$$\begin{aligned} 1 &= 7 - 3 \cdot 2 \\ &= 7 - 3(16 - 2 \cdot 7) = 7 \cdot 7 - 3 \cdot 16 \\ &= 7(55 - 3 \cdot 16) - 3 \cdot 16 = 7 \cdot 55 - 24 \cdot 16 \\ &= 7 \cdot 55 - 24(126 - 2 \cdot 55) = 55 \cdot 55 - 24 \cdot 126. \end{aligned}$$

So 55 is it's own inverse modulo 126. The decoding function is therefore $D(x) \equiv 55(x-12) \pmod{126}$, or $D(x) \equiv 55x + 116 \pmod{126}$.

6. Amalgamated signs the contract $x$ by applying $D_a$ to $x$ to obtain $D_a(x)$, and then sends $E_z(D_a(x))$ to the lawyers. Likewise, Behemoth sends them $E_z(D_b(x))$. Both messages are secure from prying eyes as only the lawyers can unlock them with $D_z$. They then apply $E_a$ to $D_a(x)$ and $E_b$ to $D_b(x)$ to verify that both are signing the correct contract $x$. If this is indeed the case, they will put $E_b$ on $D_a(x)$ and send it to Behemoth, and put $E_a$ on $D_b(x)$ and send it to Amalgamated.

7. We already know that if the message is divided into 3 parts, we need nine couriers. With only eight available, we divide the message into A, B, C and D. The couriers carry (A,B), (A,B), (A,C), (B,C), (C), (D), (D) and (D) respectively. To get all four parts, one of the last three must be a double agent, but the enemy will not get all of A, B and C with only one other double agent.

8. With six couriers, we divide the message into A, B, C, D, E and F. A working scheme has them carrying (A,B,C), (A,B,D), (C,D,E), (C,D,F), (E,F,A) and (E,F,B) respectively. Suppose we divide the message into A, B, C, D and E only. Then there are 15 parts in all. Hence some courier must carry at least 3 parts. Certainly, he should not carry 4 or more. Hence we may assume that he carries A, B and C. We have 6 D and E parts to be carried by the other five couriers, so that one of them must carry both of them. Hence it is possible for the enemy to get the complete message if two couriers are double agents.

## B.4 Recursion & Induction

1. We have

$$\begin{aligned}
a_n &= 3a_{n-1} - 1 \\
&= 3(3a_{n-2} - 1) - 1 \\
&= 3^2(3a_{n-3} - 1) - 3 - 1 \\
&= 3^3(3a_{n-4} - 1) - 3^2 - 3 - 1 \\
&= \cdots \\
&= 3^n - (3^{n-1} + 3^{n-2} + \cdots + 3 + 1) \\
&= 3^n - \frac{1}{2}(3^n - 1) \\
&= \frac{1}{2}(3^n + 1).
\end{aligned}$$

2. Denote the desired sum by $S$. Subtracting $S$ from $3S$, we have

$$\begin{array}{rrcrcrcrcccrcr}
 & 3S & = & & & 3 & + & 3^2 & + & \cdots & + & 3^n & + & 3^{n+1}, \\
-) & S & = & 1 & + & 3 & + & 3^2 & + & \cdots & + & 3^n, & & \\
\hline
 & 2S & = & -1 & & & & & & & & & + & 3^{n+1}.
\end{array}$$

It follows that $S = \frac{1}{2}(3^{n+1} - 1)$.

3. Note that $(x+1)^4 = x^4+4x^3+6x^2+4x+1$. Letting $x = 1, 2, \ldots, n$ in turn, we have

$$\begin{array}{rrrrrrr}
 & 2^4 = & 1^4 + & 4\cdot 1^3 + & 6\cdot 1^2 + & 4\cdot 1 + & 1, \\
 & 3^4 = & 2^4 + & 4\cdot 2^3 + & 6\cdot 2^2 + & 4\cdot 2 + & 1, \\
 & 4^4 = & 3^4 + & 4\cdot 3^3 + & 6\cdot 3^2 + & 4\cdot 3 + & 1, \\
 & \cdots = & \cdots + & \cdots + & \cdots & + \cdots + & \cdots, \\
 & n^4 = & (n-1)^4 + & 4(n-1)^3 + & 6(n-1)^2 + & 4(n-1) + & 1, \\
+) & (n+1)^4 = & n^4 + & 4n^3 + & 6n^2 + & 4n + & 1, \\
\hline
 & (n+1)^4 = & 1 + & 4S_3 + & 6S_2 + & 4S_1 + & n.
\end{array}$$

Here, $S_1 = \frac{1}{2}n(n+1)$ from Example 2, $S_2 = \frac{1}{6}n(n+1)(2n+1)$ from Example 6, and $S_3$ is the desired sum. From the last equation, we have $S_3 = \frac{1}{4}n^2(n+1)^2$.

4. Denote the desired sum by $S$. Subtracting from it $\frac{1}{2}S$, we have

$$\begin{array}{rrrrrrrr}
 & S = & \frac{1}{1} + & \frac{2}{2} + & \frac{3}{4} + & \cdots + & \frac{n+1}{2^n}, & \\
-) & \frac{1}{2}S = & & \frac{1}{2} + & \frac{2}{4} + & \cdots + & \frac{n}{2^n} + & \frac{n+1}{2^{n+1}}, \\
\hline
 & \frac{1}{2}S = & 1 + & \frac{1}{2} + & \frac{1}{4} + & \cdots + & \frac{1}{2^n} - & \frac{n+1}{2^{n+1}}.
\end{array}$$

Hence $S = 2(1+\frac{1}{2}+\frac{1}{4}+\cdots+\frac{1}{2^n}) - \frac{n+1}{2^n} = 4\big(1-\frac{1}{2^{n+1}}\big) - \frac{n+1}{2^n} = 4 - \frac{n+3}{2^n}$.

5. Let $a_n$ denote the number of pairings with $2n$ players. Clearly, $a_1 = 1$. In the general case, pick out any participant. Since she has to play, her opponent can be chosen from any of the remaining $2n-1$ players. Once we set this pair aside, the remaining participants can be paired in $a_{n-1}$ ways. Hence $a_n = (2n-1)a_{n-1} = (2n-1)(2n-3)a_{n-2} = \cdots = (2n-1)(2n-3)\cdots 3a_1 = (2n-1)(2n-3)\cdots 1$ as desired.

6. (a) Clearly, $a_0 = 0$. For $n \geq 1$, we divide the task into three stages. The middle stage consists of the single move of the largest disk. It has to be moved at some point, and there is no point in moving it more than once. It can be moved only when all the other disks are out of the way, together in one of the two remaining spaces.

   (b) Now the first stage consists of moving this stack of $n-1$ disks, and this takes $a_{n-1}$ moves. The last stage consists of moving this stack back on top of the largest disk, and this also takes $a_{n-1}$ moves. It follows that $a_n = 2a_{n-1}+1$.

   (c) $a_n = 2^n - 1$, $n = 0, 1, 2, \ldots$.

7. The general rule is $1^2 - 2^2 + 3^2 - \cdots + (-1)^{n-1}n^2 = (-1)^{n-1}(1+2+3+\cdots+n)$. For the basis $n = 1$, the left side is $1^2 = 1$ and the right side is $(-1)^0(1) = 1$. As the induction hypothesis, we assume that the result holds for some $n = \ell$. To complete the inductive argument, we must show that the result also holds for $n = \ell+1$. We

have

$$\begin{aligned}
1^2 - 2^2 + 3^2 - \cdots + (-1)^\ell(\ell+1)^2 &= (-1)^{\ell-1}\frac{1}{2}\ell(\ell+1) + (-1)^\ell(\ell+1)^2 \\
&= (-1)^\ell\frac{1}{2}(\ell+1)(-\ell+2(\ell+1)) \\
&= (-1)^\ell\frac{1}{2}(\ell+1)(\ell+2) \\
&= (-1)^\ell(1+2+3+\cdots+(\ell+1)).
\end{aligned}$$

8. Note that $2^1 = 2 < 3 = 2\cdot 1 + 1$ and $2^2 = 4 < 5 = 2\cdot 2+1$ but $2^3 = 8 > 7 = 2\cdot 3+1$. We claim that $2^n > 2n+1$ for all $n \geq 3$. The basis $n = 3$ has been verified. Suppose $2^\ell > 2\ell+1$ for some $\ell \geq 3$. Then $2^\ell > 2$. Adding these two inequalities yields $2^{\ell+1} = 2^\ell + 2^\ell > 2\ell+1+2 = 2(\ell+1)+1$, which completes the inductive argument.

9. Note that $2^1 = 2 > 1 = 1^2$. Clearly, $2^2 = 2^2$ and we also have $2^4 = 16 = 4^2$. However, $2^3 = 8 < 9 = 3^2$ while $2^5 = 32 > 25 = 5^2$. We claim that $2^n > n^2$ for $n = 1$ and all $n \geq 5$. The isolated case $n = 1$ and the basis $n = 5$ have been verified. Suppose $2^\ell > \ell^2$ for some $\ell \geq 5$. By Exercise 7, $2^\ell > 2\ell+1$. Adding these two inequalities yields $2^{\ell+1} = 2^\ell + 2^\ell > \ell^2+2\ell+1 = (\ell+1)^2$, which completes the inductive argument.

10. We have $a_1 = 2^1 - 1 = 1$, which agrees with the initial condition. Suppose $a_\ell = 2^\ell - 1$ for some $\ell \geq 1$. From the recurrence relation, $a_{\ell+1} = 2a_\ell + 1 = 2(2^\ell - 1)+1 = 2^{\ell+1}-1$, which completes the inductive argument.

11. We make a table to to help us guess:

| number of cards (n): | 3 | 4 | 5 | 6 | 7 | 8 | 9 | $\cdots$ |
|---|---|---|---|---|---|---|---|---|
| value of last card: | 2 | 4 | 2 | 4 | 6 | 8 | 2 | $\cdots$ |

We observe that if there are $n = 2^m$ cards, the last card seems to be card number $n$. If there are $n = 2^m + k$ cards, where $1 < k < 2^m$, the last card seems to be card number $2k$, and we make this our conjecture.

Base cases: The table shows it is true for the cases $n = 3, 4, \ldots, 9$.

Induction step: Assume that the conjecture is true for $n = 1, 2, \ldots p$. We will show it is true for $n = p+1$.

Consider the fact that when you have put a card on the table, and placed the next card on the bottom, the situation is that you have $p$ cards to be dealt, and in your hand they are now numbered in the following order:

$$3, 4, 5, 6, \ldots, p-1, p, 2$$

That is, card #2 is a 3, card #2 is a 4, and so on.

We now consider two cases: If $p = 2^m$, then the bottom card, namely the 2, will be dealt by the induction hypothesis. So the conjecture is proved when $n = p+1$ for the case $p = 2^m$.

If $p = 2^m + k$, then by the induction hypothesis, the card in position $2k$ will be the last one dealt: But the card in position $2k$ is the card with number $2(k+1)$. So the conjecture is true if $n = p + 1$ for the case where $p = 2^m + k$, $k = 1, 2, \ldots, 2^m - 1$. Note that when $k = 2^m - 1$, the number $n$ becomes $2^{m+1}$, so this completes the proof.

12. The inductive step fails when $k = 1$. When you remove horse $h_k$ from this set of size $k = 1$, there are none left, so when you replace it with $h_{k+1}$, that is, with $h_2$, you have a set containing only $h_2$ and you cannot now conclude anything about its colour.

## B.5 Graph Theory

1. Construct a graph with 20 vertices representing the teams. Two teams which square off against each other in the first day are joined by a red edge, and those in the second day by a blue edge. Each vertex is of degree 2, and the graph is partitioned into cycles. Moreover, each cycle has an even number of edges since they must be alternately red and blue. We take every other vertex in every cycle. This gives us 10 vertices no two of which are joined by an edge. They represent the 10 teams we seek.

2. Construct a graph $G$ with the vertices representing the cities, and edges representing direct flights. All vertices have degree 10 except for the vertex $C$ representing the capital city, which has degree 100. Delete all 100 edges incident with $C$ and consider a component $H$ of the resulting graph $G'$. Each vertex has degree 10 unless it was connected to $C$, in which case its degree is 9. Since $G$ is connected, $H$ has at least one vertex of degree 9. By the Parity Theorem, it must have an even number of such vertices. It follows that $H$ has at least two vertices of degree 9. Since there are exactly 100 vertices of degree 9 in $G'$, it has at most 50 components. We can obtain a connected graph $G''$ by putting back one edge to $C$ from each component of $G'$. However, $G''$ may be obtained directly from $G$ by deleting 50 edges incident with $C$.

3. The host can get by with 17 bags, with contents of 7, 7, 7, 7, 7, 7, 7, 4, 4, 4, 4, 3, 3, 3, 1, 1 and 1. To prove that 17 is indeed minimum, construct a bipartite graph with 7 vertices $x_i$, $1 \leq i \leq 7$ on one side and 11 vertices $y_j$, $1 \leq j \leq 11$ on the other. Suppose we have a set of bags which will work whether 7 or 11 children turn up. Each bag is given to a child represented by some $x_1$ in the first scenario, and to a child represented by some $y_j$ in the second scenario. We represent this bag by an edge joining $x_i$ to $y_j$. Consider a component $H$ of the graph. The total number of marbles in the bags represented by its edges must be a multiple of 7, since each $y$-vertex in $H$ represents a child who gets 7 marbles. Similarly, this total is also a multiple of 11. Since the least common multiple of 7 and 11 is 77, $H$ is the whole graph. In other words, the graph is connected. Since it has 18 vertices, it must have at least 17 edges.

4. (a) The solution by induction:

   Base case: When $n = 2$. Then the two piles are each of size 1, and the sum of the products is 1 which equals $2(2-1)/2$.

   Induction hypothesis: Assume the statement is true for $n = 2, 3, \ldots, k$.

Induction step: Split the pile of $k+1$ pennies into piles of size $m$ and $k+1-m$. This product is $m(k+1-m)$. By the induction when the splitting process is carried out on the piles of size $m$ and $k+1-m$, the totals are, respectively, $m(m-1)/2$ and $(k+1-m)(k-m)/2$. Then the sum of all the products is

$$\begin{aligned} m(k+1-m) + \frac{m(m-1)}{2} + \frac{(k+1-m)(k-m)}{2} \\ &= \frac{2m(k+1-m) + m(m-1) + (k+1-m)(k-m)}{2} \\ &= \frac{2m(k+1) - 2m^2 + m^2 - m + (k+1)k - (k+1)m - mk + m^2}{2} \\ &= \frac{(k+1)k}{2} \end{aligned}$$

(b) Using Graph Theory:

Consider the complete graph $K_n$. It has $n(n-1)/2$ edges.

Let us consider erasing these edges as follows: First split the vertices into two groups of size $m$ and $k$, and erase every edge that connects a vertex from the $m$ group with a vertex from the $k$ group. We have just erased $mk$ edges, and this leaves us with the two complete graphs $K_m$ and $K_k$.

Now repeat the process with $K_m$ and $K_k$. At each stage the number of edges erased is the product as defined. Eventually, we will have erased all edges, and no edge gets erased twice, so the sum of the products will be $n(n-1)/2$.

5. We apply Kruskal's Algorithm as summarized in the following chart.

| Vertices → | $A$ | $B$ | $C$ | $D$ | $E$ | $F$ |
|---|---|---|---|---|---|---|
| Edges ↓ | 1 | 2 | 3 | 4 | 5 | 6 |
| $CD$ | 1 | 2 | 3 | 3 | 5 | 6 |
| $BC$ | 1 | 2 | 2 | 2 | 5 | 6 |
| $DE$ | 1 | 2 | 2 | 2 | 2 | 6 |
| $AB$ | 1 | 1 | 1 | 1 | 1 | 6 |
| $EF$ | 1 | 1 | 1 | 1 | 1 | 1 |

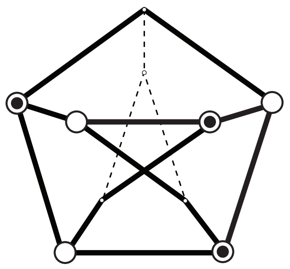
6. We apply Dijkstra's Algorithm as summarized in the following chart:

| $A$ | $B$ | $C$ | $D$ | $E$ | $F$ | ← Vertices |
|---|---|---|---|---|---|---|
| $-,0$ | $-,\infty$ | $-,\infty$ | $-,\infty$ | $-,\infty$ | $-,\infty$ | ↓ Edges |
| | $A,3$ | $A,7$ | $-,\infty$ | $-,\infty$ | $-,\infty$ | $AB$ |
| | | $B,5$ | $B,8$ | $B,7$ | $-,\infty$ | $BC$ |
| | | | $C,6$ | $B,7$ | $-,\infty$ | $CD$ |
| | | | | $B,7$ | $D,14$ | $BE$ |
| | | | | | $E,10$ | $EF$ |

7. Fix the number $V$ of vertices of the graph. Suppose $E = 0$. Then the graph consists of isolated vertices, each of which constitutes a component. Hence $C = V$. On the other hand, $F = 1$ since the infinite plane is not subdivided. It follows that $V + F = E + C + 1$. Suppose the formula holds for planar graphs with $E = \ell$ edges. Consider a planar graph with $E = \ell + 1$ edges. Delete any edge. By induction hypothesis, we have $V + F = E + C + 1$. Now restore this edge. Then $E$ goes up by 1. If it connects two vertices in the same component, then $C$ is unchanged, but $F$ goes up by 1 since an existing face is divided into two. If it connects two vertices in different components, then $C$ goes down by 1 while $F$ is unchanged. In either case, we still have $V + F = E + C + 1$, completing the inductive argument. To obtain Euler's Formula for connected planar graphs, just note that $C = 1$ so that $V - E + F = 2$.

8. Suppose the Petersen graph is planar. Since $V = 10$ and $E = 15$, we must have $F = 7$. They have among them a total of 30 boundaries. We will have a contradiction if we can show that each face must have at least 5 boundaries. Denote each of the five outside vertices by $A$ and each of the five inside vertices by $B$. We do not have triangles or quadrilaterals whose vertices are all $A$'s or all $B$'s. Suppose $AB$ is a side of a face. Then this $A$ vertex is connected to another $A$, and this $B$ vertex is connected to another $B$, but those two are not adjacent. It follows that each face has at least 5 boundaries as desired.

9. The Petersen graph contains a subdivision of $K_{3,3}$ and is non-planar by Kuratowski's Theorem. This forbidden subgraph is shown in the diagram in dark edges, with the three vertices on each side marked in distinct symbols.



10. Choose any vertex. It is adjacent to at most 3 vertices, each of which is adjacent to at most 2 others, for a total of 10. There cannot be any other vertices as such vertices are at a distance more than 2 from the chosen vertex. The Petersen graph shows that 10 is indeed the maximum.

## B.6 Digraph Theory

1. Each of $A$, $C$ and $F$ can beat each of $B$, $D$ and $E$ one on one. Since $A$ beats $C$, $C$ beats $F$ and $F$ beats $A$, each of these three may be the eventual winner.

2. Let $A$ be the vertex with the largest out-degree. If its in-degree is 0, then its radius is 1, and so is that of the tournament. Suppose there is an arc leading from some vertex $B$ to $A$. Consider all vertices accessible from $A$ in one step. If all of them are also accessible from $B$ in one step, then the out-degree of $B$ will be higher than that of $A$, which contradicts our maximality assumption on $A$. Hence there is a vertex $C$ with an arc going from $A$ to $C$, and another arc from $C$ to $B$. It follows that the radius of $A$ is 2, and so is that of the tournament.

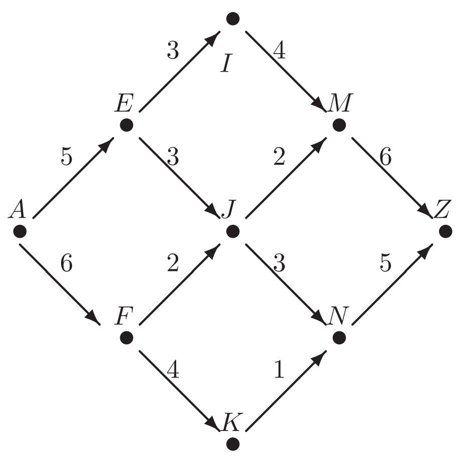
3. As in the solution to Exercise 2, the vertex $A$ with the largest out-degree must be a centre. Consider the sub-tournament consisting of all vertices with arcs going from them to $A$. Since $A$ is of radius 2, this sub-tournament is non-empty. Let $B$ be the vertex with the largest out-degree within it. Now all vertices here are accessible from $B$ in at most two steps. All vertices outside this sub-tournament are also accessible from $B$ in two steps via $A$, except for $A$ itself which is accessible from $B$ directly. It follows that $B$ is also a centre. Consider now the sub-tournament consisting of all vertices with arcs going from them to $B$, and let $C$ be the one with the largest out-degree here. Then $C$ is also a centre of the whole tournament. Moreover, $C \neq A$ since an arc goes from $C$ to $B$ and another arc goes from $C$ to $B$. If we repeat this argument, the new candidate may turn out to be $A$, and this will be the case if the whole tournament consists of just three vertices.

4. An application of the critical-path algorithm yields the following chart:

| $K$ | $A$ | $B$ | $C$ | $D$ | $E$ | $F$ | $X$ |
|---|---|---|---|---|---|---|---|
| $t(K)$ | 0 | 3 | 0 | 6 | 6 | 8 | 9 |
| $T(K)$ | 0 | 3 | 2 | 8 | 6 | 8 | 9 |
| $s(K)$ | 0 | 0 | 2 | 2 | 0 | 0 | 0 |

The critical tasks are $A, B, E$ and $F$, and they form a critical path leading to $X$.

5. We create an artificial source $A$ with arcs going to $E$ and $F$. Their capacities are the productivities of the factories. We also create an artificial sink $Z$ with arcs coming from $M$ and $N$. Their capacities are the orders of the markets. The modified transportation network is shown in the diagram below.

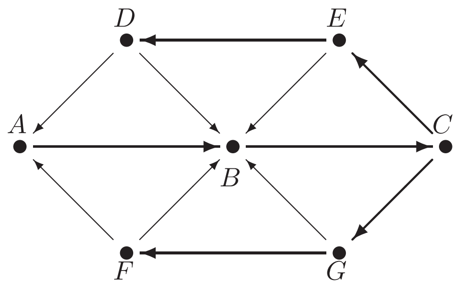


An application of the Ford-Fulkerson Algorithm yields the following chart.

| $A$ | $E$ | $F$ | $I$ | $I$ | $K$ | $M$ | $N$ | $Z$ | Path | Flow |
|---|---|---|---|---|---|---|---|---|---|---|
| $-,\infty$ | $A,5$ | $A,6$ | $E,3$ | $E,3$ | $F,4$ | $I,4$ | $I,3$ | $M,6$ | $EIM$ | 3 |
| $-,\infty$ | $A,2$ | $A,6$ | $-,0$ | $E,3$ | $F,4$ | $J,2$ | $J,3$ | $M,3$ | $EJM$ | 2 |
| $-,\infty$ | $-,0$ | $A,6$ | $-,0$ | $F,2$ | $F,4$ | $-,0$ | $J,3$ | $N,5$ | $FJN$ | 3 |
| $-,\infty$ | $-,0$ | $A,4$ | $-,0$ | $-,0$ | $F,4$ | $-,0$ | $K,1$ | $N,3$ | $FKN$ | 1 |
| $-,\infty$ | $-,0$ | $A,3$ | $-,0$ | $-,0$ | $F,3$ | $-,0$ | $-,0$ | $-,0$ | | |

We can send 5 units of goo to $M$ and 3 units to $N$.

6. We list the vertices in alphabetical order. The orientation produced by Robbins' Algorithm is shown in the diagram below.



7. We compute the radii of the individual vertices in the following chart.

| | $A$ | $B$ | $C$ | $D$ | $E$ | $F$ | $G$ | Radius |
|---|---|---|---|---|---|---|---|---|
| $A$ | 0 | 1 | 2 | 4 | 3 | 4 | 3 | 4 |
| $B$ | 4 | 0 | 1 | 3 | 2 | 3 | 2 | 4 |
| $C$ | 3 | 2 | 0 | 2 | 1 | 2 | 1 | 3 |
| $D$ | 1 | 1 | 2 | 0 | 3 | 4 | 3 | 4 |
| $E$ | 2 | 1 | 2 | 1 | 0 | 4 | 3 | 4 |
| $F$ | 1 | 1 | 2 | 4 | 3 | 0 | 3 | 4 |
| $G$ | 2 | 1 | 2 | 4 | 3 | 1 | 0 | 4 |

The diameter of the digraph is 4, the radius 3 and the centre $C$.

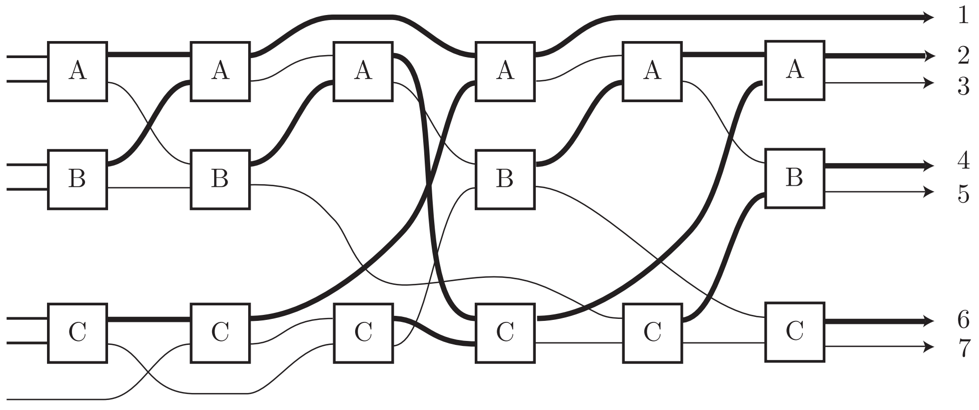
## B.7 Miscellaneous Topics

1. In the first 4 hours, have the players go against one another in pairs. In the next 5 hours, get a complete ranking of the 4 first-round winners. At this point, we have the following partial ranking.

$$\begin{tikzcd}
A \arrow[r] \arrow[d] & B \arrow[r] \arrow[d] & C \arrow[r] \arrow[d] & D \arrow[r] & E \\
F & G & H & &
\end{tikzcd}$$

   Have G play against D in hour 10, and against C or E in hour 11 according to whether G beats or loses to D. Then G is incorporated into the main stream, which for now consists of A, B, C, D and E. Next we fit H in among D, E and possibly G. As we did for G, play H against the player in the middle of a line of odd length or either of the middle players in a line of even length. This takes 2 hours. Finally, we fit F in among B, C, D, E, F and G. This can be done in 3 hours, bringing the total to 16.

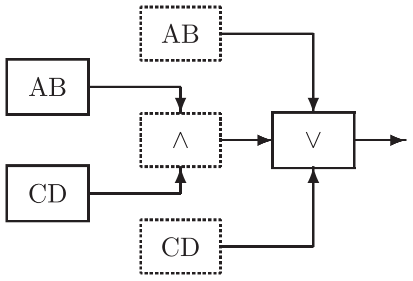
2. We obtain a solution by modifying the chart in Example 7.1.3. We add to the team a fictitious player who loses automatically. Start this player at the bottom of the chart. He will stay there and emerge as #8. Since his games do not have to be played, we only need 3 courts. The modified chart is shown below.



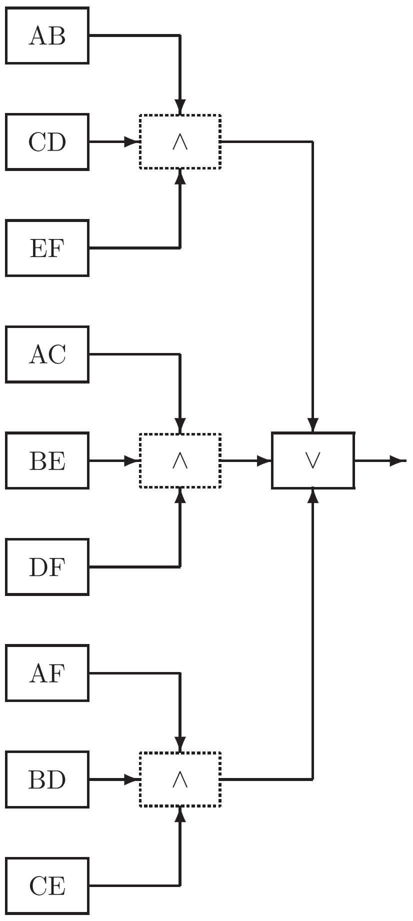
3. Consider first the second round, after which we must identify the lightest type of clamps. Hence there can only be a single multi-way comparison. With 7 tralances, we can make 21 pairwise comparisons. If there are 7 types of clamps still in contention at this point, then there are exactly 21 pairs of them. What we need is a way of placing them on the tralances so that no pair appears together more than once. This can be accomplished as follows. Let the types of clamps be A, B, C, D, E, F and G. We put the following triples on the tralances: (A,B,C), (A,D,E), (A,F,G), (B,D,F), (B,E,G), (C,D,G) and (C,E,F). In the first round, we may have 21 types of clamps, with each tralance eliminating 2 of them. Clearly, 21 is the maximum value.

4. Only one weighing is needed. Take 1 coin from the 1st stack, 2 coins from the 2nd, 3 from the 3rd, and so on. If the total weight is an integer, then the fake coins are from the 10th stack. If it is $0.1 \times k$ short of an integer, where $1 \leq k \leq 9$, then the fake coins are from the $k$-th stack.

5. The top half of the circuit in Example 7.2.1 already checks 4 pairs of signals. The bottom half double-checks 2 of them. It is only necessary to check the other 2 pairs separately. This modified circuit is shown on the right.



6. The following circuit is based on the Denham matrix $D_3$.



7. Since the first two statements of the two spies agree, they can be believed. It follows that either X and Y are true while W and Z are false, or W, Z and exactly one of X and Y are true. Hence A's third statement cannot be correct, and B is the trustworthy spy. From her third statement, W, X and Z are true while Y is false.

8. Since A claims that C is faulty, at least one of them is. Similarly, either B or F is faulty. Hence the maximum number of good units is four, which must then include D and E. D claims that F is good, and so it is, and that also makes E's claim true whether C is good or not. Since B is faulty, A must be good and C must be faulty.

9. Let the East Coast cities be $A, B$ and $C$. In the first half, one plane from each West Coast city takes all mail for $A$ and $B$. These four meet in pairs, exchanging mail for $A$ and mail for $B$. The other plane from each West Coast city takes all mail for $C$ and $D$, meet in pairs and do a similar exchange. In the second half, the planes fly to their destinations, with two planes arriving at each East Coast city.

10. We first prove that 4 minutes are needed for 5 observation posts. After 2 minutes, no observation post knows about all the attacks. In the third minute, one of the observation posts is not in communication with the others. It will not learn about all the attacks until the fourth minute. With 6 observation posts $A, B, C, D, E$ and $F$, this can be accomplished in 3 minutes. In the first minute, $A$ and $D$ exchange information while $B$ and $E$ do likewise, as do $C$ and $F$. In the second minute, the exchanges are between $A$ and $E$, $B$ and $F$, as well as $C$ and $D$. In the third minute, the exchanges are between $A$ and $F$, $B$ and $D$, as well as $C$ and $E$.

11. Let the warehouses be $A, B, C, D$ and $E$, and the barrels of toxic chemicals be 1, 2, 3, 4 and 5. In the first day, the trucks move barrels 2, 3, 4 and 5 from $A$ and $C$ to $B$ nd $D$, bringing back barrel 1. In the second day, one truck moves barrels 2 and 3 from $B$ to $D$, bringing back barrels 2 and 3. The other truck moves barrels 1, 2 and 3 from $E$ to $C$. In the third day, one truck moves barrels 3 from $B$ to $C$, bringing back barrels 2. The other truck moves barrels 5 from $D$ to $E$, bringing back barrels 4. In the fourth day, one truck takes barrels 1 from $C$ to $A$. This cannot be accomplished in three days. After the second day, no warehouse can have five barrels of the same kind, and one of them is not involved in any exchanges in the third day.

# Index