Chapter 30

Solutions to Quizes & Exercises

Do the quizzes and exercises by yourself. When you get stuck, take a peak here.

Chapter 0

Numbers and Sets.
1. 252 = 2 × 3 × 3 × 3 × 7.
2. minimum is 3.
3. union = {1, 3, 7, 8, 9, 10, 19}; intersection = {3, 10}.
4. Yes, there must be a minimum element and the minimum is at most 18.
5. 5 (integer, rational, real); 3 4 (rational, real); π (real).
6. 3k, 3k + 3.

Logarithms and Exponentials.
1. ln(12) = ln(2 × 2 × 3) = ln(2) + ln(2) + ln(3) ≈ 2.484.
2. 210 = (210)2 ≈ 1000 = 106.
3. ln(1 × 2 × 3 × · · · × 10) = (ln 1 + ln 2 + ln 3 + · · · + ln 10).
4. How are \(2^{a/b} \cdot 2^b = 2^{a-b}\).
5. By definition of \(\log_{10}\), \(100 = 10^{\log_{10} 100}\). Taking \(\log_{2}\) of both sides, \(\log_{2} 100 = \log_{2}(10^{\log_{10} 100}) = \log_{10} 100 \times \log_{2} 10\).

More generally, \(x = \beta^{\log_{\beta} x}\); taking \(\log_{\alpha}\) of both sides, \(\log_{\alpha} x = \log_{\beta} x \times \log_{\alpha} \beta\).

Sums and Products.
1. (a) 1 + 2 + 3 + · · · + 1000 = \(\frac{1}{2} \times 1000 \times 1001 = 500,500\).
   (b) 1 + 2 + 3 + · · · + n = \(\frac{1}{2}n(n + 1)\).
   (c) 1 + \(\frac{1}{7} + \frac{1}{21} + \frac{1}{35} + \frac{1}{39} + \cdots = \frac{1}{1 - \frac{1}{7}} = \frac{7}{6}\).
2. 5! = 20; \(n! = n \times (n - 1) \times (n - 2) \times \cdots \times 2 \times 1; 0! = 1\).
3. \[\sum_{i=1}^{1000} i = \sum_{k=1}^{1000} k = \frac{1}{2} \times 1000 \times 1001 = 500,500. \sum_{k=1}^{1000} i = 1000 \times i.\]
4. 1 + 2 + 3 + · · · + k = \(\sum_{i=1}^{k} i = \frac{1}{2}k(k + 1). \sum_{k=1}^{n} k = \frac{1}{2}n(n + 1).\)
5. \[\sum_{i=1}^{k} \ln(i) = \ln(k!); \prod_{i=1}^{k} i = k!\]

Algebra.
1. (1 + 2)2 = 32 = 9. Also, (1 + 2)2 = 12 + 2 × 1 × 2 + 22 = 9.
2. \((a + b)^2 = a^2 + 2ab + b^2; (a + b)^3 = a^3 + 3a^2b + 3ab^2 + b^3; (a + b)^4 \neq a^4 + 4a^3b + 4a^2b^2 + 4ab^3 + b^4.\)
3. \(x^2 - 5x - 6 = (x - 6)(x + 1) = 0, \text{ therefore the roots are } x = 6 \text{ and } x = -1\)
4. To get solutions to \( e^{2x} - 5e^x - 6 = 0 \), set \( y = e^x \); then \( y^2 - 5y - 6 = 0 \) and (using the previous problem) \( y = e^x = 6 \) or \( y = e^x = -1 \). Therefore \( x = \ln 6 \) or \( x = i\pi \) where \( i = \sqrt{-1} \). Other solutions are obtained by adding integer multiples of \( 2\pi i \).

5. \( x + y = 2 \) and \( 2x + 3y = 7 \) implies \( x = -1 \) and \( y = 3 \).

6. \( \frac{3x+11}{x^2-x-6} = \frac{4}{x-3} - \frac{1}{x+2} \) and \( \frac{3x+11}{x^3+4x^2+9} = \frac{3}{x+3} + \frac{2}{(x+3)^2} \).

**Calculus.**

1. Derivatives: \( 3x^2; \ 2x^2; \ 2^x \ln 2; \ - \frac{1}{x^2}; \ - \frac{1}{x^3}; \ - \frac{1}{x}; \ - \frac{1}{x^{10}}; \ - \frac{1}{x^2} \).

2. Integrals: \( \frac{1}{2}x^2; \ \frac{1}{3}x^2; \ \frac{1}{4}x^2; \ \ln |x|; \ - \frac{1}{x} \).

3. Limits as \( x \to 0 \):
   \[
   \frac{e^x - 1}{\sin(2x)} \xrightarrow{x \to 0} \frac{1}{2}; \ \frac{e^x - 1}{x} \xrightarrow{x \to 0} 0; \ \frac{e^x - 1}{\sin(x^2)} \xrightarrow{x \to 0} \infty; \ \frac{e^x - 1}{x + x^2} \xrightarrow{x \to 0} 1; \ \frac{e^x - 1}{e^{2x} - 1} \xrightarrow{x \to 0} \frac{1}{2}.
   \]

4. Limits as \( x \to \infty \):
   \[
   \frac{e^x - 1}{x^2} \xrightarrow{x \to \infty} 0; \ \frac{e^x - 1}{x^2 + 2x} \xrightarrow{x \to \infty} \frac{1}{2}; \ \frac{e^x}{x^2} \xrightarrow{x \to \infty} 0.
   \]

5. Which of these series converges:
   \[
   1 + 2 + 2^2 + 2^3 + 2^4 + \cdots \text{ diverges}
   \]
   \[
   1 + \frac{1}{2} + \frac{1}{2} + (\frac{1}{2})^3 + (\frac{1}{2})^4 + \cdots \text{ converges to 2}
   \]
   \[
   1 - 1 + 1 - 1 + 1 - 1 + \cdots \text{ diverges}
   \]
   \[
   1 + \frac{1}{2} + \frac{1}{2} + \cdots \text{ diverges}
   \]
   \[
   1 - \frac{1}{2} + \frac{1}{2} - \frac{1}{2} + \frac{1}{2} - \frac{1}{2} + \cdots \text{ converges to 0.6931}
   \]

6. \( f(x) = \frac{1}{x^2 + \sin(x)} = \frac{1}{2} + \frac{1}{12}(x - \frac{\pi}{2})^2 + \frac{1}{720}(x - \frac{\pi}{2})^4 + \frac{1}{34560}(x - \frac{\pi}{2})^6 + \cdots \).

7. Using the substitution \( u = \arctan(x) \), \( du = dx / (1 + x^2) \) and so \( \int_0^T dx \frac{1}{1 + t^2} = \arctan(T) \).

8. For \( f(t) = \int_0^t dx \sin(1 + x^2e^x) \), \( \frac{df}{dt} f(t) = \sin(1 + t^2e^x) \).

**Pop Quiz 0.1.** A powerful tactic when a problem looks hard is to make it easier. Suppose the letters lined up vertically. That's trivial. Now morph this simple problem into the one we want.

**Easier problem**

**Modify: move A**

**Modify: move C**

**Solution.**

Do not underestimate the power of simplification. Be on the lookout for this technique of making a problem easier. It helps to understand a problem. It builds confidence when you can solve something. It can pinpoint where the difficulty is in the harder problem. We call it tinkering.

---

**Chapter 1**

**Pop Quiz 1.1.** The red square is safe. The final infection is on the right.

**Exercise 1.2.** Six infections won’t infect the whole grid; seven is the minimum.

**Chapter 2**

**Pop Quiz 2.1.** \( O = \{ n \mid n = 2k - 1; \ k \in \mathbb{N} \} \)

**Exercise 2.2.** True if the set of pigs that fly \( \mathbb{P} \) is a subset of the set of things which are green with purple spots \( \mathbb{G} \). Since \( \mathbb{P} \) is the empty set, \( \mathbb{P} \subseteq \mathbb{G} \) and so it is a true sentence.

**Pop Quiz 2.3.** \( M \cap V = \{ a, i \}; \ M \cup V = \{ m, a, e, i, k, l, o, u \} \); taking the universal set as the lower case letters, \( \mathcal{M} = \{ \text{lower case letters other than } m, a, l, i, k \} \).

**Exercise 2.4.**

(a) Visually different because the nodes and edges are drawn in different positions.

(b) The friendships are the same, so it could be the same network.

**Exercise 2.5.** (a) \( \{-1, -2, \ldots\} = \{ n \mid n = -k; \ k \in \mathbb{N} \} \) (b) \( \{1, \frac{1}{2}, \frac{1}{3}, \ldots\} = \{ r \mid r = \frac{1}{n}; \ n \in \mathbb{N} \} \)
Chapter 3

**Pop Quiz 3.1.** It is not easy to verify. You might try asking A for a dream; then see if B has that same dream; if so, ask C, and so on. If A, B, C, D, E, F all have that same dream, then you have verified the claim. If not, ask A for another dream and repeat. Either A will run out of dreams, or you will verify the claim. The process will terminate if A has a finite number of dreams.

**Exercise 3.2.** (a) F. (b) Don’t know yet. (c) Who is Kilam? (d) T.

**Exercise 3.3.** \( p \land r \): It is raining and it is cloudy (T; it can be cloudy without rain).
\( p \rightarrow q \): If it is raining then Kilam has his umbrella (T; Kilam is a smart guy).
\( p \rightarrow r \): If it is raining then it is cloudy (T; you need clouds for rain).
\( q \rightarrow r \): If Kilam has his umbrella then it is cloudy (T; why does Kilam have an umbrella?).
\( q \rightarrow p \): If Kilam has his umbrella then it is raining (F as it could just be cloudy).
\( r \rightarrow p \): If it is cloudy then it is raining (F it can be cloudy without rain).

**Exercise 3.4.**
(a) T; (i) Yes, it is cloudy. (ii) No, it is clear.
(b) F; (i) We don’t know if it is raining.
(c) T; (i) Yes. (ii) Don’t know; you could just be smart. (iii) Don’t know. (iv) No.
(d) T; (i) Yes. (ii) Don’t know; you could just be wandering. (iii) Not hungry; not thirsty.

**Pop Quiz 3.5.** In \( C^{++} \), \( \| \) is OR, and \( \& \& \) is AND. To show that both codes execute the instructions for the same \( x, y \) values, define the propositions \( p : x > 0, q : y > 1 \) and \( r : x < y \).

The left code tests \( p \lor (q \land r) \) before executing the instructions, and the right tests \( p \lor q \). We show their truth-tables (right). The highlighted row 3 is a problem. The truth values are different.

Let’s examine closer: \( p \) is \( F \), \( x \leq 0 \); \( q \) is \( T \), \( y > 1 \); and, \( r \) is \( F, x \geq y \). This row in the truth-table is impossible: \( x \leq 0 \) and \( y > 1 \) implies \( x < y \), so \( r \) is \( T \). To compare compound propositions, you only need to consider all the possible truth values of the basic propositions. If the basic propositions are independent all 8 possibilities are relevant: \( p \lor (q \land r) \) is not equivalent to \( p \lor q \) in general. In our case, \( p, q, r \) being \( F, T, F \) is not possible: our basic propositions are not independent because the truth value of \( r \) is constrained by the truth values of \( p \) and \( q \).

For all possible truth values of \( p, q, r \), the compound propositions match, so the two snippets perform identical computations. The right snippet is simpler, uses fewer operations and, if implemented in hardware, requires fewer gates. These can be important considerations in some applications.

**Exercise 3.6.**
\[ \neg p \rightarrow q \equiv q \rightarrow p \equiv p \lor q \]
\[ \neg (p \lor q) \equiv \neg p \land \neg q \]
\[ (q \land \neg r) \rightarrow \neg p \equiv (\neg p \lor \neg q) \lor r \equiv (p \land q) \rightarrow r \]
\[ p \lor (q \lor r) \equiv \neg r \rightarrow (p \lor q) \]

**Pop Quiz 3.7.** (a) \( n \in \mathbb{N} \). (b) A predicate cannot be \( T \) or \( F \). (c) “4 is a perfect square.” (d) \( P(4) \) and \( P(9) \) are \( T \).

**Exercise 3.8.**
(a) \( P(x) = \text{“} x \text{ has grey hair”} \). \( P(\text{Kilam}) \).
(b) \( P(x) = \text{“} Map x \text{ can be colored with 4 colors with adjacent countries having different colors”} \).
\( \forall x : P(x) \).
(c) \( P(n) = \text{“} Integer n is a sum of two primes.” } \forall n \in E : P(n) \) (\( E \) is set of even natural numbers).
(d) \( P(x) = \text{“} x \text{ has blue eyes and blond hair”} \). \( \neg \exists x : P(x) \).

There can be many ways to formulate a statement with predicates. Here is another way.
\( P(x) = \text{“} x \text{ has blue eyes; } Q(x) = \text{“} x \text{ has blond hair”} \). \( \neg \exists x : P(x) \land Q(x) \).

**Exercise 3.9.**
(\( \exists a : (G(a)) \land (\exists a : H(a)) \)): someone has blue eyes and someone has blond hair.
(\( \exists b : (G(b)) \land (\exists c : H(c)) \)): someone has blue eyes and someone has blond hair. A quantified statement does not change when you change the name of a (variable) parameter.
Exercise 3.10.

(a) \( \forall a : (\exists d : P(a,d)) \) : every American has every dream.
\( \forall d : (\forall a : P(a,d)) \) : Every dream is had by every American. (both are equivalent)

(ii) \( \exists a : (\exists d : P(a,d)) \) : Some American has a dream.
\( \exists d : (\exists a : P(a,d)) \) : Some dream is had by at least one American. (both are equivalent)

(b) They are valid predicates. In English:
\( Q(a) = \exists d : P(a,d) = \text{“Some dream is had by American } a\text{”} \)
\( R(d) = \forall a : P(a,d) = \text{“All Americans have dream } d\text{”} \)

Exercise 3.11. (a) is easier to disprove. To disprove (a), you just need to find a single instance of \( n \) for which \( 2^{2^n} + 1 \) is not prime. To disprove (b), you need to show that for every choice of \( (a, b, c), a^3 + b^3 \neq c^3 \). Disproving a “there exists” is typically harder than disproving a “for all.” This is related to the fact that proving a “for all” is harder than proving a “there exists”. This is because \( \neg \exists x : P(x) \equiv \forall x : \neg P(x) \), so showing that a “there exists” statement is false means you are showing that a “for all” statement is true. Similarly \( \neg \forall x : P(x) \equiv \exists x : \neg P(x) \).

Chapter 4

Pop Quiz 4.1.

(a) \( p : n \text{ is greater than } 2 \text{ and even} \)
(b) \( p : x \text{ and } y \text{ are rational} \)
(c) \( p : ax^2 + bx + c = 0 \text{ and } a \neq 0 \)

Exercise 4.2.

(a) Proof. We use a direct proof.
1. Assume that \( a \) is divisible by \( b \) and \( b \) is divisible by \( c \).
2. This means there are integers \( k, \ell \) for which \( a = kb \) and \( b = \ell c \).
3. Then, \( a = kb = k\ell c = mc \), where \( m = k\ell \).
4. Since \( m = k\ell \) is an integer, \( a \) is divisible by \( c \), as was to be shown.

(b) A proof does not have to be written in algorithmic steps.

Proof. Let \( x \) and \( y \) be arbitrary real numbers. First observe that \( \pm x \leq |x| \) and \( \pm y \leq |y| \). There are two cases: (i) \( x + y \geq 0 \), in which case \( |x + y| = x + y \leq |x| + |y| \) (because \( x \geq 0 \) and \( y \leq |y| \)). (ii) \( x + y < 0 \), in which case \( |x + y| = -(x + y) = -x - y \leq |x| + |y| \) (because \( -x \leq |x| \) and \( -y \leq |y| \)). In both cases \( |x + y| \leq |x| + |y| \).

(c) Proof. Consider any four consecutive integers, and let \( x \) be the minimum, so the four integers are \( x, x + 1, x + 2, x + 3 \). One of these four numbers must be divisible by 4, and so equals 4k. Among the remaining numbers, two are consecutive so one is divisible by 2 and so equals 2\ell. Therefore the product of all four numbers is \( 4k \times 2\ell \times (\text{integer}) \), which is a multiple of 8.

The proof uses subtle reasoning. We leave it to the reader to give a more detailed proof based on cases. Consider the remainder when \( x \) is divisible by 4 (the remainder is either 0,1,2,3).

So there are 4 cases: \( x = 4k, x = 4k + 1, x = 4k + 2, x = 4k + 3 \). Show that in each of the four cases, the product is divisible by 8. For example, if \( x = 4k \) then the product is \( 4k(4k + 1)(4k + 2)(4k + 3) = 8k(4k + 1)(2k + 1)(4k + 3) \).

Pop Quiz 4.3. You need to find one \( n^* \in D \) for which \( Q(n^*) \) is \( F \). It is equivalent to disproving the implication: if \( n \in D \), then \( Q(n) \).

Exercise 4.4.

(a) The two truth-tables are identical. The only case \( p \rightarrow q \) is \( F \) with \( p \ T \) and \( q \ F \). The only case \( \neg q \rightarrow \neg p \) is with \( \neg q \ T \) and \( \neg p \ F \), i.e. with \( p \ T \) and \( q \ F \). The only case

(b) (i) If the grass is not wet, THEN it did not rain last night.
(ii) If \( x \leq 10 \) and \( y \leq 10 \), then one of \( x, y \) is not positive or \( xy \leq 100 \).
Proof. (By contraposition) Suppose $x \leq 10$ and $y \leq 10$. There are two cases.
Case 1: One of $x, y$ is not positive in which case “one of $x, y$ is not positive or $xy \leq 100$” is true and there is nothing further to prove.
Case 2: Both $x, y$ are positive, so $0 < x, y < 10$. In this case $xy < 10 \times 10 = 100$ and so “one of $x, y$ is not positive or $xy \leq 100$” is true.

(iii) If you do not stay at home, then the mall is not crowded.
(iv) If $\sqrt{r}$ is rational then $r$ is rational.

Proof. (By contraposition) Suppose $\sqrt{r}$ is rational. Then $\sqrt{r} = a/b$ for integer $a$ and natural number $b$. This means $r = a^2/b^2$ which is rational because $a^2$ is an integer and $b^2$ is a natural number.

Pop Quiz 4.5. The truth-tables are the same: $p \leftrightarrow q \equiv (p \rightarrow q) \land (q \rightarrow p)$ (logically equivalent).

Exercise 4.6.
(a) (Do not intersect, but not parallel according to the true definition.)
(b) Two line segments (in 3-dimensions) are parallel if and only if they both lie in the same plane and when both are extended to infinity in both directions, there is no point of intersection.
(c) A triangle is isosceles if and only if at least two sides have the same length.

Exercise 4.7.
(a) To get a contradiction, suppose there are $m, n \in \mathbb{Z}$ with $21m + 9n = 3(7m + 3n) = 1$. $3$ divides the LHS, therefore $3$ divides $1$. FISHY! This contradiction proves the claim.
(b) Suppose $x, y > 0$ and $x + y < 2\sqrt{xy}$. Both sides of the inequality are positive. Squaring, $(x + y)^2 < 4xy$, or $x^2 + 2xy + y^2 < 4xy$, or $x^2 - 2y + y^2 < 0$ or $(x - y)^2 < 0$. This is FISHY because the square of a real number cannot be negative. This contradiction proves the claim.
(c) Suppose that $m$ and $n$ are both odd. Then $m = 2k + 1$ and $n = 2\ell + 1$. $m^2 + n^2 = 2k^2 + 4k + 1 + 4\ell^2 + 4\ell + 1$. Since $m^2 + n^2$ is divisible by $4$, $m^2 + n^2 = 4s$, therefore $m^2 + n^2 = 4(k^2 + k + \ell^2 + \ell) + 2$, or $4(s - k^2 - k - \ell^2 - \ell) = 2$ or $2(s - k^2 - k - \ell^2 - \ell) = 1$. The LHS is divisible by $2$ therefore $1$ is divisible by $2$, FISHY. This contradiction proves the claim.

Exercise 4.8.
(a) Direct proof because the result clearly follows from the assumption that $x$ is real.
(b) Contraposition, for if $n$ is even (not odd), then it is easy to show by algebra that $n^2$ is odd.
(c) Direct proof because by simple algebra if $n$ is odd one can square and show $n^2$ is even.
(d) Show an example of a number that is not a square and prove it.
(e) Direct proof because, by simple algebra, a product of two ratios is a ratio.
(f) Direct proof. By simple algebra one can show that a product of two odd numbers is odd.
(g) Contradiction. This gives you something to start with. You assume that a rational $p/q$ equals $\sqrt{6}$. The contradiction will show that this rational does not exist.
(h) This does not seem to fall into a standard situation. When you can’t see where to start, often contradiction is your best bet, because a proof by contradiction gives you something to work with. Let us illustrate how contradiction can be used to prove this non-trivial result.

Proof. Let $x_1, \ldots, x_n$ be arbitrary numbers and let $\mu = (x_1 + \cdots + x_n)/n$ be the average, so that $x_1 + \cdots + x_n = n\mu$. Now assume that every $x_i < \mu$ (to obtain a contradiction). Then $x_1 + \cdots + x_n < \mu + \cdots + \mu = n\mu$. This is a contradiction. Therefore, not every $x_i < \mu$, so at least one number is as large (or larger) than the average $\mu$.

This is a very important fact; commit it to memory. There is always a number that is at least as large as the average. Does there have to be at a number that is larger than the average?

Pop Quiz 4.9. We prove that $x \in A \cap B \rightarrow x \in C$ using direct proof. Assume $x \in A \cap B$. Then $x \in A$ and $x \in B$, so $x$ is even and $x = 9k$. If $k$ is odd, then $x$ is the product of two odd numbers which is odd. Therefore, $k$ is even (to make $x$ even). So, $k = 2n$, which means $x = 18n = 6 \cdot (3n)$, which is a multiple of $6$. Therefore, $x \in C$, which concludes the proof.

Exercise 4.10.
Exercise 5.2.

(a) \((A \cap B) \cup (A \cap C) = A \cap (B \cup C)\):

Suppose \(x \in (A \cap B) \cup (A \cap C)\). Then either \(x \in A \cap B\) or \(x \in A \cap C\), and so \(x \in A \cap (B \cup C)\). Now suppose \(x \in A \cap (B \cup C)\). Then \(x \in A\) and either \(x \in B\) or \(x \in C\). In both cases, \(x \in (A \cap B) \cup (A \cap C)\).

(b) \(A \cup \overline{B} = \overline{A \cap B}\):

Suppose \(x \in A \cup \overline{B}\). This means \(x \in A\) or \(x \in \overline{B}\). In both cases, \(x \in A \cap \overline{B}\). Now suppose \(x \in \overline{A \cap B}\). That is, \(x \in A \cup \overline{B}\), so \(x \notin A \cap B\).

Chapter 5

Exercise 5.1.

(a) \(S(n)\) is a sum of integers so it is an integer, call it \(k\).

(b) By the high-school geometric sum formula: \(1 + 4 + 4^2 + \cdots + 4^{n-1} = (4^n - 1)/(4 - 1) = (4^n - 1)/3\).

(c) Therefore \(k = (4^n - 1)/3\), or \(4^n - 1 = 3k\). That is, \(4^n - 1\) is divisible by \(3\).

Exercise 5.2. (a) \(n \geq 2\) (b) \(n \geq 0\) (c) \(n = 0, 1\) (d) \(n \geq 1\) (e) \(n \geq 1\)

Exercise 5.3.

(a) Define the claim \(P(n) : \sum_{i=1}^{n-1} a + \text{id} = na + \frac{1}{2}n(n-1)d\).

1. \([\text{Base case}]\) \(P(1)\) claims that \(a = a\), which is clearly \(\top\).

2. \([\text{Induction step}]\) We show \(P(n) \rightarrow P(n+1)\) for all \(n \geq 1\), using a direct proof.

   Assume (induction hypothesis) \(P(n)\) is \(\top\): \(\sum_{i=1}^{n-1} a + \text{id} = na + \frac{1}{2}n(n-1)d\).

   Show \(P(n+1)\) is \(\top\): \(\sum_{i=1}^{n} a + \text{id} = \sum_{i=1}^{n-1} a + \text{id} + a + \text{id} = a + nd + na + \frac{1}{2}n(n-1)d = (n+1)a + \frac{1}{2}(n(n+1))d = (n+1)a + \frac{1}{2}(n+1)nd\).

   (IH stands for “by the induction hypothesis”). We have shown \(P(n+1)\) is \(\top\), as needed.

3. By induction, \(P(n)\) is \(\top\) \(\forall n \geq 1\).

(b) Define the claim \(P(n) : \sum_{i=1}^{n-1} ar^i = a(r^n - 1)/(r - 1)\).

1. \([\text{Base case}]\) \(P(1)\) claims that \(a = a\), which is clearly \(\top\).

2. \([\text{Induction step}]\) We show \(P(n) \rightarrow P(n+1)\) for all \(n \geq 1\), using a direct proof.

   Assume (induction hypothesis) \(P(n)\) is \(\top\): \(\sum_{i=1}^{n-1} ar^i = a(r^n - 1)/(r - 1)\).

   Show \(P(n+1)\) is \(\top\): \(\sum_{i=1}^{n} ar^i = \sum_{i=1}^{n-1} ar^i + ar^n = a(r^n - 1)/(r - 1) + ar^n = a(r^{n+1} - 1)/(r - 1)\).
We compute the sum $\sum_{i=1}^{n} ar^i$ as follows:
$$\sum_{i=1}^{n} ar^i = ar^n + nd + \sum_{i=1}^{n-1} ar^i \equiv ar^n + \frac{a(r^{n+1} - 1)}{r-1} = \frac{a(r^{n+1} - 1)}{r-1}.$$

We have shown that $P(n+1)$ is T, as needed.

3: By induction, $P(n)$ is T $\forall n \geq 1$.

(c) Define the claim $P(n) : n \leq 2^n$.
1: [**Base case**] $P(1)$ claims that $1 \leq 2^1$, which is clearly T.
2: [**Induction step**] We show $P(n) \rightarrow P(n+1)$ for all $n \geq 1$, using a direct proof.
   Assume (induction hypothesis) $P(n)$ is T: $n \leq 2^n$. 
   Show $P(n+1)$ is T: $n+1 \leq 2^{n+1}$.
   $$n+1 \leq 2^n + 1 \leq 2^n + 2^n = 2^{n+1}.$$

We have shown that $P(n+1)$ is T, as needed.

3: By induction, $P(n)$ is T $\forall n \geq 1$.

(d) Define the claim $P(n) : 5^n - 1$ is divisible by 4.
1: [**Base case**] $P(1)$ claims that $5 - 1$, is divisible by 4, which is clearly T.
2: [**Induction step**] We show $P(n) \rightarrow P(n+1)$ for all $n \geq 1$, using a direct proof.
   Assume (induction hypothesis) $P(n)$ is T: $5^n - 1$ is divisible by 4, so $5^n - 1 = 4k$.
   Show $P(n+1)$ is T: $5^{n+1} - 1$ is divisible by 4.
   $$5^{n+1} - 1 = 5 \cdot 5^n - 1 \equiv 5 \cdot (4k+1) - 1 = 20k + 4 = 4(5k+1).$$

Therefore $5^{n+1} - 1$ is divisible by 4, and we have shown that $P(n+1)$ is T.

3: By induction, $P(n)$ is T $\forall n \geq 1$.

(e) Define the claim $P(n) : \sum_{i=1}^{n} i \cdot i! = (n+1)! - 1$.
1: [**Base case**] $P(1)$ claims that $1 = 2! - 1$, which is clearly T.
2: [**Induction step**] We show $P(n) \rightarrow P(n+1)$ for all $n \geq 1$, using a direct proof.
   Assume (induction hypothesis) $P(n)$ is T: $\sum_{i=1}^{n} i \cdot i! = (n+1)! - 1$.
   Show $P(n+1)$ is T: $\sum_{i=1}^{n+1} i \cdot i! = (n+2)! - 1$. We compute $\sum_{i=1}^{n+1} i \cdot i!$ as follows:
   $$\sum_{i=1}^{n+1} i \cdot i! = (n+1)(n+1)! + \sum_{i=1}^{n} i \cdot i! \equiv (n+1)(n+1)! + (n+1)! - 1 = (n+1)! (n+1+1) - 1 = (n+2)! - 1.$$

We have shown that $P(n+1)$ is T, as needed.

3: By induction, $P(n)$ is T $\forall n \geq 1$.

**Pop Quiz 5.4.** The claim is readily verified by substituting $a_0, a_1, a_2, a_3$ into the four equations.

**Exercise 5.5.**

(a) Tinker. Compute $S(n)$ for small n: $n$ | 1 2 3 4 5 6 7 8 9 10 ... 
$S(n)$ | 1 4 9 16 25 36 49 64 81 100 ...

A reasonable guess is $S(n) = n^2$. The proof by induction follows the standard template. For
the base case, $S(1) = 1^2$. Suppose $S(n) = n^2$ and consider
$$S(n+1) = S(n) + 2n + 1 \equiv n^2 + 2n + 1 = (n+1)^2.$$

($\equiv$ stands for “by the induction hypothesis”). By induction, $S(n) = n^2$ for all $n \geq 1$.

(b) As usual, first tinker with small n: $n$ | 1 2 3 4 5 6 ... 
$S(n)$ | 1 9 36 100 225 441 ... 
$\sum_{i=1}^{n} i$ | 1 3 6 10 15 21 ...
A reasonable guess is $S(n) = (\sum_{i=1}^{n} i)^2$. The proof by induction follows the standard template. For the base case, $S(1) = 1 = 1^2$. Suppose $S(n) = (\sum_{i=1}^{n} i)^2$ and consider

$$S(n + 1) = S(n) + (n + 1)^3 = (\sum_{i=1}^{n} i)^2 + (n + 1)^3 = \frac{1}{4}n^2(n + 1)^2 + (n + 1)^3,$$

where the last step follows from the formula $\sum_{i=1}^{n} i = \frac{1}{2}n(n + 1)$. Therefore,

$$S(n) = \frac{1}{4}(n + 1)^2(n^2 + 4(n + 1)) = \frac{1}{4}(n + 1)^2(n + 2)^2.$$

The last expression is $(\sum_{i=1}^{n+1} i)^2$. By induction, $S(n) = (\sum_{i=1}^{n} i)^2$ for all $n \geq 1$.

**Pop Quiz 5.6.** (a) $n \geq 3$ (b) $n = 3, 4$ (c) $n \geq 1$

**Exercise 5.7.**

(i) Define the set $C = \{x + z_0 \mid x \in B\}$. Then $C$ contains only natural numbers, and is non-empty because $B$ is non-empty. By the well-ordering principle $C$ has a minimum element $c_0 = b_0 + z_0$ where $b_0 \in B$. Consider any $b \in B$. Then $c = b + z_0 \in C$ and therefore $c_0 \leq c$, i.e. $b_0 + z_0 \leq b + z_0$ or $b_0 \leq b$. This proves that $b_0$ is a minimum element of $B$.

(ii) (a) Let $n_*$ be the smallest counter-example; $n_* \geq 2$. Then $n_* \geq 2$ (P(1) is t). Therefore $\sum_{i=0}^{n_*-1} a + id \neq n_*a + \frac{1}{2}n_*(n_*-1)d$ and, because $n_*$ is the smallest counter-example, $\sum_{i=0}^{n_*-2} a + id = (n_*-1)a + \frac{1}{2}(n_*-1)(n_*-2)d$. But,

$$\sum_{i=0}^{n_*-1} a + id = a + (n_*-1)d + \sum_{i=0}^{n_*-2} a + id$$

$$= a + (n_*-1)d + (n_*-1)a + \frac{1}{2}(n_*-1)(n_*-2)d$$

$$= n_*a + \frac{1}{2}n_*(n_*-1)d,$$

which contradicts $n_*$ being a counter-example: there is no counter-example.

(b) Let $n_*$ be the smallest counter-example; $n_* \geq 2$. Then $n_* \geq 2$ (P(1) is t) and $n_* > 2^{n_*}$. Also, $n_* - 1$ is not a counter-example (because $n_*$ is the smallest counter-example), so $n_* - 1 \leq 2^{n_*}$. But,

$$n_* = n_* - 1 + 1 \leq 2^{n_*} + 1 \leq 2^{n_*} + 2^{n_*} = 2^{n_*},$$

which contradicts $n_*$ being a counter-example: there is no counter-example.

(c) Let $n_*$ be the smallest counter-example: $n_* \geq 2$. Then $n_* \geq 2$ (P(1) is t) and $n_* > 2^{n_*}$. Also, $n_* - 1$ is not a counter-example (because $n_*$ is the smallest counter-example), so $n_* - 1 \leq 2^{n_*}$. But,

$$n_* = n_* - 1 + 1 \leq 2^{n_*} + 1 \leq 2^{n_*} + 2^{n_*} = 2^{n_*},$$

which contradicts $n_*$ being a counter-example: there is no counter-example.

(d) Let $n_*$ be the smallest counter-example: $n_* \geq 2$. Then $n_* \geq 2$ (P(1) is t) and $5^{n_*} - 1$ is not divisible by 4. Also, $n_* - 1$ is not a counter-example (because $n_*$ is the smallest), so $5^{n_*} - 1 = 4k$. But,

$$5^{n_*} - 1 = 5 \cdot 5^{n_*-1} - 1 = 5(4k + 1) - 1 = 4(5k + 1),$$

so 4 divides $5^{n_*} - 1$ contradicting $n_*$ being a counter-example: $n_*$ does not exist.

(e) Let $n_*$ be the smallest counter-example: $n_* \geq 2$. Then $n_* \geq 2$ (P(1) is t). Therefore $\sum_{i=1}^{n_*} i! \neq (n_* + 1)! - 1$. Since $n_*$ is the smallest counter-example: $\sum_{i=1}^{n_*-1} i! = n_*! - 1$. But,

$$\sum_{i=1}^{n_*} i! = n_*n_*! + \sum_{i=1}^{n_*-1} i! = n_*n_*! + n_*! - 1 = (n_* + 1)! - 1,$$

which contradicts $n_*$ being a counter-example: there is no counter-example.

**Exercise 5.8.** Suppose $P(1)$ is t; and, $P(n) \rightarrow P(n + 1)$ is t for $n \geq 1$. We show $P(n)$ is t for all $n \geq 1$. Assume $P(n)$ is false for some $n$, and let $n_*$ be the smallest counter-example for $P(n)$; $n_* \geq 2$ because $P(1)$ is t. Therefore $n_* - 1$ is not a counter-example (because $n_*$ is the smallest), so $P(n_* - 1)$ is t. But, $P(n_* - 1) \rightarrow P(n_*)$, since $n_* - 1 \geq 1$ and since $P(n_* - 1)$ is t, it implies that $P(n_*)$ is t. This contradicts $n_*$ being a counter-example, so $P(n)$ is t for all $n \geq 1$.

**Exercise 5.9.** Suppose $B$ is a non-empty subset of $\mathbb{N}$. Suppose $B$ has no minimum element. We show that $B$ is empty (a contradiction). Define the claim

$$P(n) : B \text{ does not contain } 1, 2, \ldots, n.$$
Therefore $R \geq n$ for $n \geq 1$. If you define “$P(n)$: $B$ does not contain $n$,” then the induction $P(n) \rightarrow P(n+1)$ fails because to show that $n+1 \notin B$ you need to know that $1, 2, \ldots, n \notin B$.

Chapter 6

Pop Quiz 6.1. Assume (to get a contradiction) that $2\sqrt{n} + \frac{1}{\sqrt{n+1}} \geq 2\sqrt{n+1} + 1$. Multiplying by $\sqrt{n+1}$ and rearranging, $2\sqrt{n(n+1)} > 2n + 1$. Since both sides are positive, we can square to obtain $4n^2 + 4n > 4n^2 + 4n + 1$, or $0 > 1$, a clear contradiction. Therefore $2\sqrt{n} + \frac{1}{\sqrt{n+1}} \geq 2\sqrt{n+1} + 1$.

Exercise 6.2. Define the claim $P(n): n^3 < 2^n$. Let us consider the induction step, so assume that $P(n)$ is $T$ and consider $(n+1)^3 = n^3 + 3n^2 + 3n + 1 < 2^n + 3n^2 + 3n + 1$. $P(n+1)$ will follow if $3n^2 + 3n + 1 < 2^n$, so define $Q(n): 3n^2 + 3n + 1 < 2^n$. Let us consider the induction step for $Q$: assume $Q(n)$, i.e. $3n^2 + 3n + 1 < 2^n$ and consider $Q(n+1)$. $3(n+1)^2 + 3(n+1) + 1 = 3n^2 + 3n + 1 + 6n + 6 < 2^n + 6n + 6$. $Q(n+1)$ will be true if $6n + 6 < 2^n$. Let us define the claim $R(n): 6n + 6 < 2^n$. Let us prove the stronger claim $P(n) \land Q(n) \land R(n)$ for $n \geq 10$.

For the base case, the reader can easily verify that $P(10)$, $Q(10)$. $R(10)$ are all $T$. For the induction step, assume $P(n) \land Q(n) \land R(n)$ for $n \geq 10$, so $n^3 < 2^n \land 3n^2 + 3n + 1 < 2^n \land 6n + 6 < 2^n$. We prove $P(n+1) \land Q(n+1) \land R(n+1)$.

$$n + 1^3 = n^3 + 3n^2 + 3n + 1 < 2^n + 2^n = 2^{n+1}$$

$$3(n+1)^2 + 3(n+1) + 1 = 3n^2 + 3n + 1 + 6n + 6 < 2^n + 2^n = 2^{n+1}$$

$$6(n+1) + 6 = 6n + 6 + 1 < 2^n + 6 < 2^n + 2^n = 2^{n+1}.$$  

The 1st equation uses $P(n)$ for $n^3$ and $Q(n)$ for $3n^2 + 3n + 1$. The 2nd equation uses $Q(n)$ for $3n^2 + 3n + 1$ and $R(n)$ for $6n + 6$. The 3rd equation uses $R(n)$ for $6n + 6$ and $6 < 2^n$ when $n \geq 10$. Therefore $P(n+1) \land Q(n+1) \land R(n+1)$ is $T$. By induction $P(n) \land Q(n) \land R(n)$ is $T$ for $n \geq 10$.

Pop Quiz 6.3. No because each subgrid has $4^n$ squares, which is not a multiple of 3 (the number of squares in an $L$-tile). This is so because we know that $4^n - 1$ is divisible by 3, so $4^n$ can’t be.

Exercise 6.4. We prove a stronger claim by induction.

$P(n):$ the $2^n \times 2^n$ grid can be $L$-tiled for any choice of the blacked out square.

The base case is the $2 \times 2$ square. For the induction step, assume $P(n)$, so the $2^n \times 2^n$ grid can be $L$-tiled for any choice of the blacked out square. We $P(n+1)$. Consider the $2^{n+1} \times 2^{n+1}$ grid with any square missing. Divide the grid into its 4 sub-squares as in the text. Place an $L$-tile in the center overlapping with the 3 sub-grids that are empty. You now have four $2^n \times 2^n$ sub-grids, each with a square missing somewhere; three of them have a corner square missing and one has a square missing in some arbitrary position. By the induction hypothesis, each of these sub-grids with a square missing can be independently $L$-tiled. Therefore the whole $2^{n+1} \times 2^{n+1}$ grid with the square missing can be $L$-tiled, and so $P(n+1)$ is $T$. By induction, $P(n)$ is true for $n \geq 1$.

Pop Quiz 6.5.

(a) $P(1)$, $P(2)$, $P(3)$, $P(4)$, $P(5)$, $P(6)$, $P(7)$, $P(8)$, $P(9)$, $P(10)$, $P(11)$, $P(12)$, $\ldots$

(b) There are three sets of implication arrows: black (starting from 1); gray (starting from 2); and, light gray (starting from 3). To touch every $n$ with a chain of implication arrows that starts from a base case, we need the three bases cases highlighted in boxes, i.e. $P(1)$, $P(2)$, $P(3)$.

Exercise 6.6.

(a) $P(1)$, $P(2)$, $P(3)$, $P(4)$, $P(5)$, $P(6)$, $P(7)$, $P(8)$, $P(9)$, $P(10)$, $P(11)$, $P(12)$, $\ldots$

A chain starts at every odd $n$ (we have shown the chains starting at 1, 3, 5).

(ii) There is no in-arrow into odd $n$, so you need infinitely many base cases 1, 3, 5, 7, 9, 11, $\ldots$.
(b) (i) The base case is $1 = 2^0$. The base case is $P(1)$ and 1 is $2^0$. We use strong induction. Assume $P(1), \ldots, P(n)$, and consider $P(n + 1)$. There are two cases.

Case 1: $n$ is even. By the induction hypothesis, $n = \sum_{i \geq 1} a_i 2^i$, where $a_i = 0, 1$. It follows that $n + 1 = 2^0 + \sum_{i \geq 1} a_i 2^i$ and so $P(n + 1)$ is $\tau$.

Case 2: $n$ is odd, so $n + 1$ is even, and $1 \leq \frac{1}{2}(n + 1) \leq n$, and so $P(\frac{1}{2}(n + 1))$ is $\tau$, that is $\frac{1}{2}(n + 1) = \sum_{i \geq 1} a_i 2^i$, where $a_i = 0, 1$. Therefore $n + 1 = \sum_{i \geq 1} a_i 2^{i + 1}$ and so $P(n + 1)$ is $\tau$.

In both cases, we proved $P(n+1)$, and so, by induction, $P(n)$ is $\tau$ for $n \geq 1$.

Exercise 6.7.

(a) $21 = 2^3 + 2^2 + 2^1$.

(b) Define $P(n) : n$ is a sum of distinct powers of 2. The base case is $P(1)$ and $1 = 2^0$. We use strong induction. Assume $P(1), \ldots, P(n)$, and consider $P(n + 1)$. There are two cases.

Case 1: $n$ is even. By the induction hypothesis, $n = \sum_{i \geq 1} a_i 2^i$, where $a_i = 0, 1$. It follows that $n + 1 = 2^0 + \sum_{i \geq 1} a_i 2^i$ and so $P(n + 1)$ is $\tau$.

Case 2: $n$ is odd, so $n + 1$ is even, and $1 \leq \frac{1}{2}(n + 1) \leq n$, and so $P(\frac{1}{2}(n + 1))$ is $\tau$, that is $\frac{1}{2}(n + 1) = \sum_{i \geq 1} a_i 2^i$, where $a_i = 0, 1$. Therefore $n + 1 = \sum_{i \geq 1} a_i 2^{i + 1}$ and so $P(n + 1)$ is $\tau$.

In both cases, we proved $P(n+1)$, and so, by induction, $P(n)$ is $\tau$ for $n \geq 1$.

Exercise 6.8.

(a) Define $P(n) :$ the greedy algorithm uses the fewest coins for $n$. The base case is $P(1)$ which is clearly $\tau$, since the greedy algorithm uses one 1c coin, and you cannot do better than one coin. We use strong induction, so assume $P(1), \ldots, P(n)$, and consider $n + 1$.

Let $n + 1 \geq 25$. Suppose that the optimal way to obtain $n + 1$ does not contain a quarter.

It cannot contain 3 or more dimes, as you can replace 3 dimes with a quarter and a nickel and do better. We leave it to the reader to show in a similar way that it cannot contain 2 dimes, 1 dime or 0 dimes, which results in an impossible situation. Therefore, the optimal way must contain a quarter; and then use some number of coins for $n + 1 - 25$. The greedy algorithm uses a quarter and then by the induction hypothesis, the optimal number of coins for $n + 1 - 25$, which means greedy is optimal for $n + 1$.

Suppose $10 \leq n + 1 < 25$. A similar reasoning shows there must be at least one dime, hence greedy is optimal. (If there is no dime, there aren’t 2 nickels, or 1 nickel and at least 5 pennies, or no nickel and at least 10 pennies.)

Suppose $5 \leq n + 1 < 10$. Using similar reasoning, there must be a nickel and greedy is therefore optimal. Lastly, Greedy is clearly optimal for $n + 1 < 5$.

Thus greedy is optimal for $n + 1$, and hence by induction, greedy is optimal for all $n \geq 1$.

(b) Consider the coin system with denominations \{1e, 4e, 5e\}. To make 8e, the greedy algorithm uses \{5e, 1e, 1e, 1e\}, but you only need two coins, \{4e, 4e\}.

Pop Quiz 6.9. No. Only $P(1)$ and $P(5)$ are $\tau$.

You need the base cases $P(1), P(2), P(3), P(4)$. Then, $P(1) \rightarrow P(5)$; $P(1) \land P(2) \rightarrow P(6)$; $P(1) \land P(2) \land P(3) \rightarrow P(7)$; $P(1) \land P(2) \land P(3) \land P(4) \rightarrow P(8)$; $P(1) \land P(2) \land P(3) \land P(4) \land P(4) \rightarrow P(9)$; and so on.
Chapter 7

Pop Quiz 7.1. \( f(3) \) cannot be computed. \( f(3) = f(2) + 5 = f(1) + 3 + 5 = f(0) + 1 + 3 + 5 = \ldots \). Since we don’t know any of \( f(2), f(1), f(0) \), we cannot compute \( f(3) \).

Exercise 7.2.
(a) \( f(-1) = f(0) = 0; f(1) = f(0) + 1 = 1; f(2) = f(1) + 3 = 4; f(3) = f(2) + 5 = 9 \);
(b) \( f(n) = n^2 \).

Exercise 7.3. (a) and (b) are well defined. In (c), the recursive part uses a larger value further from a base case. In (d) you cannot compute \( f(1) \).

Exercise 7.4. \( P(0) \) and \( P(1) \) are true since \( f(n) \) is explicitly given for \( n = 0, 1 \). We show that \( P(n) \Rightarrow P(n + 1) \). Assume \( f(n) \) can be computed. Then \( f(n + 2) = f(n) + 2 \) which can be computed since we know \( f(n) \). Therefore, by leaping induction, \( P(n) \) is \( \forall \) for \( n \geq 0 \).

Exercise 7.5.
(a) \( f(n) = 2n \). (Base case) \( f(0) = 0 = 2 \times 0 \). (Induction step) Assume \( f(n) = 2n \); then, \( f(n + 1) = 2 + f(n) = 2 + 2n = 2(n + 1) \).
(b) \( f(n) = 0 \). (Base case) \( f(0) = 0 \). (Induction step) Assume \( f(n) = 0 \); then, \( f(n + 1) = 2f(n) = 2 \times 0 = 0 \).
(c) \( f(n) = 2^n \). (Base case) \( f(0) = 1 = 2^0 \). (Induction step) Assume \( f(n) = 2^n \); then, \( f(n + 1) = 2f(n) = 2 \times 2^n = 2^{n+1} \).

Exercise 7.6. First we unfold the recursions in (a), (b), (c); (d) is complicated.

(a) \[
\begin{align*}
& f(n) = f(n-1) + \log_2 n \\
& f(n-1) = f(n-2) + \log_2 (n-1) \\
& f(n-2) = f(n-3) + \log_2 (n-2) \\
& \vdots
\end{align*}
\]
\[
\frac{f(3)}{3} = f(2) \times \log_2 3
\]
\[
\frac{f(2)}{2} = f(1) + \log_2 2
\]
\[
\frac{f(1)}{1} = f(0) = \log_2 n!
\]

(b) \[
\begin{align*}
& f(n) = 2f(n-1) \\
& f(n-1) = 2f(n-2) \\
& f(n-2) = 2f(n-3) \\
& \vdots
\end{align*}
\]
\[
\frac{f(3)}{3} = f(2) \times 2f(1) \\
\frac{f(2)}{2} = f(1) \times 2f(0) \\
\frac{f(1)}{1} = f(0) \times 2f(-1) = 2
\]

(c) \[
\begin{align*}
& f(n) = nf(n-1) \\
& f(n-1) = (n-1)f(n-2) \\
& f(n-2) = (n-2)f(n-3) \\
& \vdots
\end{align*}
\]
\[
\frac{f(3)}{3} = f(2) \times f(1) \\
\frac{f(2)}{2} = f(1) \\
\frac{f(1)}{1} = f(0) = n!
\]

In (a), the cancellations occur when you equate the sum of the LHS terms to that of the RHS terms. In (b) and (c), the cancellations occur when you equate the products. Here are the proofs.

(a) \( f(n) = \log_2 n! \). (Base case) \( f(1) = 0 = \log_2 1! \). (Induction step) Assume \( f(n) = \log_2 n! \); then, \( f(n + 1) = f(n) + \log_2 (n + 1) = \log_2 n! + \log_2 (n + 1) = \log_2 (n + 1)! \).

(b) \( f(n) = 2^{n-1} \). (Base case) \( f(1) = 1 = 2^0 = 2^{1-1} \). (Induction step) Assume \( f(n) = 2^{n-1} \); then, \( f(n + 1) = 2f(n) = 2 \times 2^{n-1} = 2^n = 2^{n+1-1} \).

(c) \( f(n) = n! \). (Base case) \( f(0) = 1 = 0! \). (Induction step) Assume \( f(n) = n! \); then, \( f(n + 1) = (n + 1)f(n) = (n + 1) \times n! = (n + 1)! \).

(d) It is possible to unfold the recursion, but one must be careful.

\[
\begin{align*}
& f(n) = f(n-1) \times f(n-2) \times \cdots \times f(1) \\
& f(n-1) = f(n-2) \times \cdots \times f(1) \\
& f(n-2) = f(n-3) \times \cdots \times f(1) \\
& \vdots
\end{align*}
\]
\[
\frac{f(3)}{3} = f(2) \times f(1) \\
\frac{f(2)}{2} = f(1) \\
\frac{f(1)}{1} = f(0) = n!
\]

Even though getting the formula was a little complicated, the proof by induction after you have the formula is standard.

(Base case) \( f(1) = 2 = 2^1 = 2^2 = 2^3 = \cdots = 2^n \).

(Induction step) Assume \( f(n) = 2^{n-1} \). Then,
\[
\begin{align*}
& f(n + 1) = f(n)^2 \\
& = 2^{n-1} \times 2^{n-1} \\
& = 2^{2(n-1)} \\
& = 2^n = 2^{n+1-1}.
\end{align*}
\]

There’s an easier analysis of this recursion.

449
Transforming a recursion. We take this opportunity to show you a powerful trick for analyzing recursions. Often you can transform $f(n)$ to a function $g(n)$ that is easier to analyze. The recursion for $f(n)$ will be transformed to one for $g(n)$. We use the transformation

$$g(n) = \log_2 f(n).$$

Note: $g(1) = \log_2 f(1) = 1$. Taking logs of both sides of the recursion for $f(n)$ gives

$$\log_2 f(n) = \log_2 f(n-1)^2 = 2 \log_2 f(n-1).$$

We get a recursion for $g(n)$ by replacing $\log_2 f$ with $g$,

$$g(n) = 2g(n-1).$$

We analyzed this recursion in part (b), $g(n) = 2^{n-1}$ and $f(n) = 2^{\log(n)} = 2^{2^{n-1}}$.

Exercise 7.7. $T_1 = 1 = F(2)$ and $T_2 = 2 = F_3$, so the base cases are true. We use strong induction. Suppose $T_1 = F_2, \ldots, T_n = F_{n+1}$ for $n \geq 2$. By the recursion, $T_{n+1} = T_n + T_{n-1} = F_{n+1} + F_n$ (by the induction hypothesis). But by the Fibonacci recursion, $F_{n+1} + F_n = F_{n+2}$, hence $T_{n+1} = F_{n+2}$. By induction, $T_n = F_{n+1}$ for $n \geq 1$.

Exercise 7.8. $F_1 = 1 \leq 2^1$ and $F_2 = 2 \leq 2^2$, so the base cases hold. Suppose $F_1 \leq 2^1, \ldots, F_n \leq 2^n$ for $n \geq 2$. Consider $F_{n+1} = F_n + F_{n-1} \leq 2^n + 2^{n-1}$ (by the induction hypothesis). But $2^n + 2^{n-1} \leq 2^n + 2^n = 2^{n+1}$, so $F_{n+1} \leq 2^{n+1}$. By induction, $F_n \leq 2^n$ for $n \geq 1$.

Exercise 7.9. We use induction. The base case is Big($0$) = 1 which is $2^0$. Suppose Big($n$) = $2^n$. Since $n + 1 > 0$, Big($n+1$) = $2 \cdot$ Big($n$) = $2 \cdot 2^n = 2^{n+1}$. By induction, Big($n$) = $2^n$ for $n \geq 1$.

Exercise 7.10. We use induction. The base case: $T_0 = 2 = 3 \times 0 + 2$. Suppose $T_n = 3n + 2$. Then, $T_{n+1} = T_n + 3 = 3n + 2 + 3 = 3(n + 1) + 2$. By induction, $T_n = 3n + 2$ for $n \geq 1$.

Exercise 7.11. (a) Yes (b) Yes (c) No $\left(\frac{3}{2} \rightarrow \frac{3}{2}\right)$ (d) Yes (e) No (1 is not in the set)

Exercise 7.12. Remember that in all cases, by default, nothing else is in the set.

(a) 1 $\in S$. (b) $\varepsilon, 0, 1 \in S$. (c) $\varepsilon \in S$. $x \in S \Rightarrow 3x \in S$. $x \in S \Rightarrow 0x \in S$. $x \in S \Rightarrow [x]y \in S$.

Exercise 7.13.

$$\varepsilon \quad T_1 = \varepsilon \quad T_2 = \varepsilon \quad T_1 = \bullet \quad T_2 = \bullet \quad T_1 = \bullet \quad T_2 = \bullet$$


(a) Every RFBT is an RBT. This is because the basis case for the RFBT is an RBT; and, the constructor rules are the same. However, every RBT is not an RFBT.

(b) There are no RFBTs with 6 vertices (only an odd $\neq$ of vertices is possible). 5 node RFBT:

Chapter 8

Pop Quiz 8.1. $\varepsilon \xrightarrow{x=\varepsilon} [ ] \xrightarrow{x=\varepsilon} [ ] \xrightarrow{y=\varepsilon} [ ] \xrightarrow{x=[\varepsilon]} [ ] \xrightarrow{y=[\varepsilon]} [ ]$

Exercise 8.2. The proof is by structural induction.

1. Clearly $\varepsilon$ is matched (base case).
2. For the induction step, there is only one constructor rule. Suppose $x$ and $y$ are matched.

Then $xy$ is matched and so every prefix in $[xy$ has at least one more “$|$” than “$\varepsilon$”. Inserting “$|$” anywhere in $xy$ can add at most one to “$|$”’s in some prefixes. Therefore, every prefix in $[x]y$ has at least as many “$|$” than “$\varepsilon$” and so $[x]y$ is matched.
3. By structural induction, every string in \( \mathcal{M} \) is matched.

**Exercise 8.3.**

(a) Suppose \( s \) is balanced and matched. For prefix \( s[0] \cdots s[i] \), define the excess function \( f(i) \) to be the number of \( "\cdot" \) minus the number of \( "-" \). Since \( s \) is matched, \( f(i) \geq 0 \); \( s \) must begin with \( "\cdot" \) so \( f(0) = 1 \) and \( s \) is balanced so \( f(n) = 0 \) (length \( s = n + 1 \)). Let \( i_\ast \) be the first prefix which is balanced, so \( f(i_\ast) = 0 \) and \( i_\ast \leq n \) and \( s[i_\ast] = "\cdot" \). We have decomposed \( s \) as

\[
s = [x]y,
\]

where \( x = s[1] \cdots s[i_\ast - 1] \) and \( y = s[i_\ast + 1] \cdots s[n] \) (\( x \) or \( y \) could be empty). To be done, we show that \( x \) and \( y \) are both balanced and matched. Since \( s[0] \cdots s[i_\ast] \) is balanced (\( f(i_\ast) = 0 \)), \( x \) must be balanced. And since \( s \) and \( y \) are balanced, \( y \) must be balanced.

We now show that \( x \) and \( y \) are matched. The easy case is \( y \). Suppose \( y \) is not matched. So there is some prefix \( \alpha \) of \( y \) with more \( "\cdot" \). Then \( [x]\alpha \) is a prefix of \( s \) with more \( "\cdot" \), because \( [x] \) is balanced. This contradicts \( s \) being matched, and so \( y \) is matched.

Suppose \( x \) is not matched. So, there is a prefix \( \beta \) of \( x \) with more \( "\cdot" \); \( \beta \neq x \) because \( x \) is balanced. Consider \( \beta \) which is a prefix of \( s \). \( f(\beta) \geq 0 \) because \( s \) is matched and \( \beta \) has more \( "\cdot" \); \( \beta \) has exactly one more \( "\cdot" \) than \( "\cdot" \), which means that \( f(\beta) = 0 \). But this contradicts \( s[0] \cdots s[i_\ast] \) being the first prefix that is balanced.

(b) Suppose \( s \) is a balanced and matched string that is not in \( \mathcal{M} \).

(i) By the well ordering principle, we can choose \( s \) to be the balanced and matched string of minimum length that is not in \( \mathcal{M} \).

(ii) By (a), \( s = [x]y \) where \( x, y \) are both balanced and matched.

(iii) \( x \) and \( y \) are at least 2 characters shorter than \( s \).

(iv) \( s \) has minimum length among balanced matched strings not in \( \mathcal{M} \). \( x, y \) are both balanced and matched, but shorter than \( s \). Thus \( x, y \in \mathcal{M} \).

(v) By the constructor rule, \( s = [x]y \in \mathcal{M} \), which contradicts \( s \notin \mathcal{M} \).

**Exercise 8.4.**

(a) \( N_s = N \).

(b) Structural induction with \( N_s \) is exactly strong induction:

1. (Basis) Show property \( P \) holds for 1, i.e. \( P(1) \) is \( \top \).

2. (Structural Induction) Assume \( P \) holds for \( 1, 2, \ldots, n \) and show \( P(n + 1) \) holds, i.e.

\[
P(1) \land P(2) \land \cdots \land P(n) \rightarrow P(n + 1)
\]

**Exercise 8.5.**

(a) \( x \cdot y = 0101110; x \cdot y \cdot z = 010111010110 \).

(b) \( x \cdot y \cdot z = 0111010; (x \cdot y \cdot z)^n = 011011011010 \).

(c) Let \( n = |y| \). We prove that \( (x \cdot y)^n = y^n \cdot x^n \) by strong induction on \( n = |y| \). If \( n = 0 \) (\( y = \epsilon \)), there is nothing to prove (base case). Suppose the claim holds up to \( n \geq 0 \), that is \( (x \cdot y)^n = y^n \cdot x^n \) whenever \(|y| \leq n \). Now consider any \( y \) with \(|y| = n + 1 \) and write \( y = y[n]b \) where \( b \) is a single bit and \( y[n] \) is the prefix of length \( n \). Then

\[
(x \cdot y)^n = ((x \cdot y[n]) \cdot b)^n \equiv b \cdot (x \cdot y[n])^n \equiv b \cdot y[n] \cdot x^n \equiv (y[n] \cdot b)^n \cdot x^n = y^n \cdot x^n.
\]

First we apply \( \mathbb{IH} \) to \( b \), then to \( y[n] \) and then to \( b \) again (all of which have length at most \( n \)).

(d) The base case, \( n = 2 \), is in (c). Assume the claim holds for \( n \geq 2 \) and consider \( n + 1 \):

\[
(x_1 \cdot x_2 \cdot \cdots \cdot x_n \cdot x_{n+1})^{n+1} = ((x_1 \cdot x_2 \cdot \cdots \cdot x_n) \cdot x_{n+1})^n \\
\equiv x_{n+1} \cdot (x_1 \cdot x_2 \cdot \cdots \cdot x_n)^n \\
\equiv x_{n+1} \cdot x_n^{n-1} \cdot \cdots \cdot x_1.
\]

**Pop Quiz 8.6.** \( \varepsilon \rightarrow 11 \rightarrow 0110 \rightarrow 001100 \).

A length 6 palindrome is \( xx^n \) for any \( x \) with \(|x| = 3 \). There are 8 strings of length 3, hence 8 palindromes of length 6. In general, there are \( 2^\left\lfloor n/2 \right\rfloor \) palindromes of length \( n \).

**Exercise 8.7.**

(a) We give the formal proof with numbered steps for easy reference.

1. The 3 base cases \( \varepsilon, 0, 1 \) are palindromes. (Strings of length at most 1 are palindromes.)
2: For the structural induction step, suppose we start with a palindrome \( x = x^n \). We must show that each constructor rule produces a new palindrome. Using Exercise 8.5, \((0 \times x \times 0)^n = 0^n \times x^n \times 0^n = 0 \times x \times 0\) and similarly, \((1 \times x \times 1)^n = 1^n \times x^n \times 1^n = 1 \times x \times 1\) (because \( x = x^n \)). Therefore both constructor rules produce palindromes.

3: By structural induction, every member of \( P \) is a palindrome.

(b) Consider \( s \), the shortest palindrome not in \( P \). If \( s \) starts with 0, then it must end in 0, so \( s = 0 \times x \times 0 \). Further, \( x \) must be a palindrome for \( s \) to be one. Now, \( x \) is shorter than \( s \), so since \( s \) is the shortest palindrome not in \( P \), it must be that \( x \in P \). But then the constructor rule gives that \( s = 0 \times x \times 0 \in P \), a contradiction. A similar contradiction arises if \( s = 1 \times x \times 1 \). Therefore, there is no shortest palindrome not in \( P \), i.e. every palindrome is in \( P \).

**Exercise 8.8.**

(a) We give the formal proof with numbered steps for easy reference.

1: The base case is 1 which clearly evaluates to 1 which is odd.

2: Structural induction: We consider each constructor rule separately. For rule 1, suppose \( x \in A_{odd} \) and \( x \) is odd. The constructor rule produces \( (x+1+1) \) and value \( (x+1+1) = value(x) + 2 \) which is odd because \( value(x) \) is odd. For rule 2, suppose \( x, y \in A_{odd} \) and \( x, y \) are odd. The constructor rule produces \( (x \times y) \) whose value is \( value(x) \times value(y) \), which is odd because the product of two odd numbers is odd.

3: By structural induction, the value of every member of \( A_{odd} \) is odd.

(b) \((1 + 1 + 1 + 1 + 1)\)

**Pop Quiz 8.9.** The number of links is 15. The number of vertices is 15. For any RBT, the number of links must be one less than the number of vertices. So this tree cannot be an RBT.

**Exercise 8.10.**

(a) First the size. By the recursive definition,

\[
size = 1 + size(left-subtree) + size(right-subtree) = 1 + 2 \cdot size(left-subtree),
\]

since both child-subtrees are identical. Applying the same logic to the left-child,

\[
size = 1 + 2(1 + 2 \cdot size(left-left-child)) = 1 + 2 + 4 \cdot size(left-left-child)
\]

\[
= 1 + 2 + 4 + 8 + 16 \cdot size(left-left-left-left-child) = 1 + 2 + 4 + 8 + 16 \cdot 0 = 15.
\]

Similarly, we can recursively obtain the height,

\[
height = 1 + 1 + 1 + 1 + 1 \cdot height(left-left-left-left-child) = 1 + 1 + 1 + 1 - 1 - 1 = 3.
\]

(b) Define, for an RBT \( T \), the property \( P(T) : size(T) \leq 2^{height(T)+1} - 1 \). \( P(\varepsilon) \) is \( \tau \) because \( 0 = 2^{-1} - 1 \). Now suppose that, for RBTs \( T_1 \) and \( T_2 \), \( P(T_1) \) and \( P(T_2) \) are \( \tau \). That is,

\[
size(T_1) \leq 2^{height(T_1)+1} - 1 \]

\[
size(T_2) \leq 2^{height(T_2)+1} - 1 \]

\[
\leftarrow \text{induction hypothesis}
\]

By the recursive definitions of size and height,

\[
height(T) = 1 + max(height(T_1), height(T_2))
\]

\[
size(T) = 1 + size(T_1) + size(T_2)
\]

By the induction hypothesis,

\[
size(T) \leq 1 + 2^{height(T_1)+1} - 1 + 2^{height(T_2)+1} - 1
\]

\[
= 2^{height(T_1)+1} + 2^{height(T_2)+1} - 1
\]

\[
\leq 2^{max(height(T_1), height(T_2))+1} + 2^{max(height(T_1), height(T_2))+1} - 1
\]

\[
= 2^{max(height(T_1), height(T_2))+2} - 1
\]

\[
= 2^{height(T)+1} - 1.
\]

So, \( P(T) \) is \( \tau \) and we have proved that the constructor preserves property \( P \). By structural induction, \( P(T) \) is \( \tau \) for every \( T \in RBT \).

**Chapter 9**

**Pop Quiz 9.1.** (a) \( 1+1+1=3 \). (b) \( 1+2+3=6 \). (c) \( f(i) = 1 + 1 + 1 = 3 \), so the answer is \( 3 + 3 + 3 = 9 \). (d) \( 1 + (1 + 2) + (1 + 2 + 3) = 10 \).
Pop Quiz 9.2. \( T_4(n) = 5 + \sum_{i=1}^{n} 10 = 5 + 10 \cdot \sum_{i=1}^{n} 1 \). The last sum is \( n \), so \( T_4(n) = 5 + 10n \).

Exercise 9.3. We use several common sums together with the constant and addition rules:

\[ S(n) = \sum_{i=1}^{n} (1 + 2i + 2^{i+2}) = \sum_{i=1}^{n} 1 + \sum_{i=1}^{n} 2i + \sum_{i=1}^{n} 2^{i+2} \quad \text{(addition rule)} \]

\[ = \sum_{i=1}^{n} 1 + 2 \sum_{i=1}^{n} i + 4 \sum_{i=1}^{n} 2^i \quad \text{(constant rule)} \]

\[ = n + 2 \times \frac{1}{2} n(n+1) + 4 \times (2^{n+1} - 1) \quad \text{(common sums)} \]

\[ = 2^{n+3} + n^2 + n - 8 \quad \text{(simplify)} \]

Exercise 9.4.

(a) \( T_1(n) = 2 + \sum_{i=1}^{n} \left[ 2 + \sum_{j=i}^{n} \left( 5 + \sum_{k=i}^{j} 2 \right) \right] \)

\[ = 2 + \sum_{i=1}^{n} \left[ 2 + \sum_{j=i}^{n} 5 + \sum_{k=i}^{j} 2 \right] \quad \text{(addition rule)} \]

\[ = 2 + \sum_{i=1}^{n} 2 + \sum_{j=i}^{n} \sum_{k=i}^{j} 2 = 2 + \sum_{i=1}^{n} \sum_{j=i}^{n} \sum_{k=i}^{j} 2 \quad \text{(addition rule)} \]

\[ = 2 + 2 \sum_{i=1}^{n} 1 + 5 \sum_{i=1}^{n} \sum_{j=i}^{n} j + 2 \sum_{i=1}^{n} \sum_{j=i}^{n} \sum_{k=i}^{j} 1 = 2 + 2 \sum_{i=1}^{n} \sum_{j=i}^{n} j + 5 \sum_{i=1}^{n} \sum_{j=i}^{n} j \quad \text{(constant rule)} \]

In the last expression we used the constant rule 3 times to pull the constants outside (to the left) in all summations.

(b) \( \sum_{k=i}^{n} 1 = j + 1 - i \) is one of the common sums.

(c) \( \sum_{i=1}^{n} (j + 1 - i) = 1 + 2 + \cdots + (n + 1 - i) = \sum_{i=1}^{n} \ell = \frac{1}{2} (n + 1 - i)(n + 2 - i) \)

(In the last expression, we used a common sum).

(d) To compute \( \sum_{i=1}^{n} (n + 1 - i)(n + 2 - i) \), we observe that as \( i \) goes from 1 up to \( n \), \( n + 1 - i \) goes from \( n \) down to 1. Letting \( \ell = n + 1 - i \), our sum can equivalently be written

\[ \sum_{\ell=1}^{n} (\ell)(\ell + 1) = \sum_{\ell=1}^{n} \ell^2 + \sum_{\ell=1}^{n} \ell. \]

(We used the addition rule to get the last expression.)

(e) We use the nested sum rule to compute \( \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} 1 \).

\[ = \sum_{i=1}^{n} \sum_{j=1}^{n} (j + 1 - i) \quad \text{(nested sum rule and (b))} \]

\[ = \frac{1}{2} \sum_{i=1}^{n} (n + 1 - i)(n + 2 - i) \quad \text{(nested sum rule and (c))} \]

\[ = \frac{1}{2} \sum_{i=1}^{n} i^2 + \frac{1}{2} \sum_{i=1}^{n} i \quad \text{(using (d))} \]

\[ = \frac{1}{2} n(n+1)(2n+1) + \frac{1}{2} n(n+1) \quad \text{common sums} \]

Whenever you compute a huge sum like this it is advisable to tinker and see if your formula is right for small values of \( n \). You can write a function in your favorite programming language to compute the sum and test it against your formula. We did exactly that to verify our formula.

\[
\begin{array}{|c|ccccccc|}
\hline
n & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
\hline
\sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} 1 & 1 & 4 & 10 & 20 & 35 & 56 & 84 \\
\frac{1}{2} n(n+1)(2n+1) + \frac{1}{2} n(n+1) & 1 & 4 & 10 & 20 & 35 & 56 & 84 \\
\hline
\end{array}
\]

Pop Quiz 9.5. Using the formulas in the table on page 115, \( T_1 \in \Theta(n^3) \); \( T_2 \in \Theta(n^2) \); \( T_3 \in \Theta(n \log n) \); \( T_4 \in \Theta(n) \).

Therefore, \( T_1 \) is in \( \Omega(n \log n) \), \( \omega(n \log n) \), \( \Omega(n^2) \), \( \omega(n^2) \), \( O(n^3) \), \( \Theta(n^3) \), \( \Omega(n^3) \).
Exercise 9.6.

(a) \( f + f = 2f \in \Theta(f) \) because 2 is a constant. Similarly \( f + f + f = 3f \in \Theta(f) \). With \( n \) terms, \( f + f + \cdots + f = nf \in \Theta(nf) \) (you cannot ignore the \( n \) because it is not a constant).

(b) \( \lim(c \cdot f) / c = c \text{ constant} \) so \( c \cdot f \in \Theta(f) \).

(c) Follows from the calculus fact that for any \( \epsilon, k > 0 \), \( \lim n \to \infty \log k n / n^\epsilon = 0 \).

(d) Follows from \( n^k / n^\log n = n^{k - \log n} \to 0 \) and \( n^k / 2^\log n \leq 2^k \log n \to 0 \).

(e) Follows from \( \log n^k = k \log n \) so \( \log n^k / \log n \to k \) is constant.

(f) Follows from \( (1 + \sqrt{n})/n = 1/n + 1/\sqrt{n} \to 0 \).

(g) Follows from \( (\frac{a}{n} + \frac{b}{n})/n = 1 + 5/n \to 1 \) is constant.

(h) We must prove upper and lower bounds. The upper bound follows from:

\[
\log n! = \log n + \log(n-1) + \cdots + \log 1 \leq \log n + \log n + \cdots + \log n = n \log n.
\]

For the lower bound, observe that \( \log(2n)! = \log(2n)(2n-1)(2n-2)(2n-3) \cdots 2 \cdot 1 \), hence

\[
\log(2n)! \leq \log(2n)^2(2n-1)^2 \cdots 2 \log n + 2 \log 2.
\]

(We get this bound by grouping in pairs, for example \( 2n(2n-1) \leq (2n)^2 \).) Also,

\[
\log(2n)! = \log(2n) + \log(2n-1) + \cdots + \log(n+1) + \log n
\]

\[
\geq \log(2n) + \log(2n-1) + \cdots + \log(n+1) + n \log n
\]

Combining the two bounds, \( 2 \log n! + 2n \log 2 \geq n \log n \), or,

\[
\log n! \geq \frac{1}{2}n \log n - n \log 2 = \frac{1}{2}n \log n + \frac{1}{2}n(\log n - 4 \log 2).
\]

We conclude that \( \log n! \geq \frac{1}{2}n \log n \) which is true from the inequality above for \( n \geq 2^4 \), because \( \log(n - 4 \log 2) \geq 0 \) and it can be verified for \( n = 1, \ldots, 15 \) explicitly.

(i) \( f = a_k n^k + g \) where \( g \) has only lower order terms, at most \( k \) such terms. Let the largest coefficient in \( g \) be \( A \). Then \( |g| \leq k |A| n^{k-1} \). We have that \( f/n^k = a_k + g/n^k \) and \( |g/n^k| \leq |A|k n^{k-1}/n^k = |A|k/n^2 \). Since \( |A| \) and \( k \) are constants, \( |g/n^k| \to 0 \) and we have that \( f/n^k \to a_k = \text{constant} \). This proves \( f \in \Theta(n^k) \).

(j) Yes, \( f \) is polynomial. The highest power “appearing” is \( n \) which has order 1. From the previous problem you might think \( f \in \Theta(n) \), but that is wrong. The notation is deceiving because there are many terms and the term of order \( n \) does not appear just once as in a traditional polynomial. For example \( n/2 \) appears somewhere in the middle, which is also of order 1. In fact, there are \( n/2 \) terms that are at least \( n/2 \), so \( f \geq n^2/4 \). In fact, we know that \( f(n) = \frac{1}{2}n(n+1) \in \Theta(n^2) \).

To make part (i) more precise, we should emphasize that in a polynomial, each term of a particular order appears at most once.

(k) Suppose \( n^2 \in O(n) \), i.e. \( n^2 \leq Cn \) for a constant \( C \), by taking \( \lceil C \rceil \), we may assume \( C \) is an integer. Let \( n = 2C \); then \( 4C^2 \leq 2C^2 \) or \( 4 \leq 2 \), a contradiction. Therefore \( n^2 \not\in O(n) \).

(l) Since \( f \in \Theta(r) \) and \( g \in \Theta(s) \), there are positive constants \( c, C \) and \( d, D \) for which

\[
c \cdot r \leq f \leq C \cdot r \quad \text{and} \quad d \cdot s \leq g \leq D \cdot s.
\]

(i) Adding the left hand sides and similarly the right hand sides gives

\[
cr + ds \leq f + g \leq Cr + Ds.
\]

Since \( cr + ds \geq \min(c, d)(r + s) \) and \( Cr + Ds \leq \max(C, D)(r + s) \), we have that

\[
\min(c, d)(r + s) \leq f + g \leq \max(C, D)(r + s),
\]

or that \( f + g \in \Theta(r + s) \).

(ii) Instead of adding, if we multiply, we get \( cd \cdot (rs) \leq f g \leq CD \cdot (rs) \), or that \( fg \in \Theta(rs) \).

(m) (i) No. Consider \( f = 2n \) and \( g = n \), then \( f \in \Theta(g) \). But \( 2^n/2^n - 2^{n^2}/2^{n^2} = 2^n \to \infty \).

(ii) Yes. We have that \( c \cdot g \leq f \leq C \cdot g \). Everything is positive and log is increasing so take log of both sides to get \( \log g + \log c \leq \log f \leq \log g + \log C \). That is \( \log f \in \Theta(\log g) \).

O-notation blurs small differences (constants). Exponentiation blows up those differences, so one must be careful. Logarithms further reduce differences and so are safe with O-notation.

454
(n) (a) \( f \in \Theta(g) \rightarrow f \in O(g) \) is \( \cap \) because \( \Theta(\cdot) \) requires upper and lower bounds but \( O(\cdot) \) requires only the upper bound: \( \Theta(g) \subseteq O(g) \). (b) The converse, \( f \in O(g) \rightarrow f \in \Theta(g) \) is \( \neg \). For a counter-example, consider \( f = n \) and \( g = n^2 \). (c) Yes: \( c \cdot g \leq f \leq c' \cdot g \) implies \( \frac{1}{c} \cdot f \leq g \leq \frac{1}{c'} \cdot f \).

(o) Suppose \( f \in O(n) \), then \( f \leq Cn \leq Cn^2 \). That is \( f \in O(n^2) \), which means \( O(n) \subseteq O(n^2) \). It is a proper subset because \( n^2 \notin O(n) \) but \( n^2 \in O(n^2) \).

\( \Theta(n) \not\subseteq \Theta(n^2) \) because \( n \in \Theta(n) \) but \( n \notin \Theta(n^2) \).

(p) We can use the definitions based on the limits or the more formal definitions based on bounds.

(i) \( f/h = (f/g) \cdot (g/h) \); since both terms on the RHS converge to a constant because \( f \in \Theta(g) \) and \( g \in \Theta(h) \), \( f/h \rightarrow \) constant, i.e. \( f \in \Theta(h) \).

(ii) \( f/h = (f/g) \cdot (g/h) \rightarrow 0 \); both terms on the RHS converge to 0, i.e. \( f \in o(h) \).

(iii) \( f \leq C \cdot g \) and \( g \leq C' \cdot h \) implies \( f \leq C \cdot (C' \cdot h) = CC' \cdot h \), i.e. \( f \in O(h) \).

(iv) \( f/h = (f/g) \cdot (g/h) \rightarrow \infty \); both terms on the RHS converge to \( \infty \), i.e. \( f \in \omega(h) \).

(v) \( f \geq C \cdot g \) and \( g \geq C' \cdot h \) implies \( f \geq C \cdot (C' \cdot h) = CC' \cdot h \), i.e. \( f \in \Omega(h) \).

(q) We need a basic identity for any pair of positive numbers \( x, y \): \( \max(x, y) \leq x + y \leq 2 \max(x, y) \).

Suppose \( r \in O(f + g) \). Then,

\[ r \leq C(f + g) \leq 2C \max(f, g), \]

i.e., \( r \in O(\max(f, g)) \) and \( O(f + g) \subseteq O(\max(f, g)) \). Suppose \( r \in O(\max(f, g)) \). Then,

\[ r \leq C \max(f, g) \leq C(f + g), \]

i.e., \( r \in O(f + g) \) and \( O(\max(f, g)) \subseteq O(f + g) \). \( O(f + g) \subseteq O(\max(f, g)) \) and \( O(\max(f, g)) \subseteq O(f + g) \) implies \( O(f + g) = O(\max(f, g)) \).

Similarly, suppose \( r \in O(\Theta(f + g)) \). Then,

\[ c \max(f, g) \leq c(f + g) \leq r \leq C(f + g) \leq 2C \max(f, g), \]

i.e., \( r \in \Theta(\max(f, g)) \) and \( \Theta(f + g) \subseteq \Theta(\max(f, g)) \). Suppose \( r \in \Theta(\max(f, g)) \). Then,

\[ \frac{1}{c}c \max(f, g) \leq c \max(f, g) \leq r \leq C \max(f, g) \leq C(f + g), \]

i.e., \( r \in \Theta(f + g) \) and \( \Theta(\max(f, g)) \subseteq \Theta(f + g) \). \( \Theta(f + g) \subseteq \Theta(\max(f, g)) \) and \( \Theta(\max(f, g)) \subseteq \Theta(f + g) \) implies \( \Theta(f + g) = \Theta(\max(f, g)) \).

(r) We have that \( c \cdot g \leq f \leq C \cdot g \). Summing: \( c \cdot \sum_i g(i) \leq \sum_i f(i) \leq C \cdot \sum_i g(i) \), which means \( \sum_i f(i) \in \Theta(\sum_i g(i)) \).

(s) We prefer \( T_1 \) because its running time is asymptotically faster.

(t) We don’t know because \( T_2 \) could be \( n \) or \( n^2 \) (both are in \( O(n^2) \)). For the former, we prefer \( T_2 \) and for the latter we prefer \( T_1 \). Whenever possible, you should give runtimes using \( \Theta \)-notation, because \( O(\cdot) \) is more ambiguous. (Just as if \( T_1 = 10 \) and \( T_2 = 20 \), which is better?)

(u) Similar to (t). \( T_2 \) could be \( n \) or \( n^{2.5} \). (Just as if \( T_1 = 10 \) and \( T_2 = 20 \), which is better?)

(v) \( T_1 \) is asymptotically better than \( T_2 \) (\( T_1 \) is “equal to” \( n^2 \) versus \( T_2 \) is “greater than” \( n^2 \)) so we definitely prefer \( T_1 \).

(w) \( T_1 \) is asymptotically no worse than \( T_2 \) (\( T_1 \) is “equal to” \( n^2 \) versus \( T_2 \) is “at least” \( n^2 \)) so we prefer \( T_1 \), though \( T_2 \) could be as good.

(x) We don’t know because \( T_2 \) could be \( n \) or \( n^3 \), both are in \( \Omega(n) \). Like \( O(\cdot), \Omega(\cdot) \) is ambiguous. Whenever possible, try to give a \( \Theta \)-analysis.

(y) \( T_2 \) is asymptotically better? Theoretically, \( T_2 \) is a better run time but see (z).

(z) Asymptotically \( T_2 \) is better. This means for \( n \rightarrow \infty, T_2 < T_1 \). But, \( T_2 \) does not become better until \( n > 10^{100} \). That is a large input, unlikely to be ever seen in practice – most estimates put the number of atoms in the Universe at fewer than \( 10^{100} \).

Exercise 9.7.

![Exercise 9.7](image)

Exercise 9.8. In all cases, let \( S(n) \) denote the sum.
(a) Since \((1+i)^2\) is increasing, 
\[
\int_0^n dx (1+x)^2 \leq S(n) \leq \int_1^{n+1} dx (1+x)^2.
\]
Computing the integrals, 
\[
\frac{1}{2}(n+1)^3 \leq S(n) \leq \frac{1}{2}((n+2)^3-1).
\]
Since the lower and upper bounds are in \(\Theta(n^3)\), \(S(n) \in \Theta(n^3)\).

(b) Since \(2^i\) is increasing, 
\[
\int_0^n dx 2^i \leq S(n) \leq \int_1^{n+1} dx 2^i.
\]
Computing the integrals, 
\[
\frac{1}{\ln 2}(2^n - 1) \leq S(n) \leq \frac{1}{\ln 2}(2^{n+1} - 2).
\]
Since the lower and upper bounds are in \(\Theta(2^n)\), \(S(n) \in \Theta(2^n)\).

(c) Since \(\sqrt{i^2}\) is increasing, 
\[
\int_0^n dx x^{2i} \leq S(n) \leq \int_1^{n+1} dx x^{2i}.
\]
Computing the integrals, 
\[
\frac{1}{\ln 2}n^{2i} - \frac{1}{\ln 2}2^{2i} + \frac{1}{\ln 2} \leq S(n) \leq \frac{1}{\ln 2}(n+1)^{2i} - \frac{1}{\ln 2}2^{2i+1} - \frac{2}{\ln 2} + \frac{2}{\ln 2}.
\]
Since the lower and upper bounds are in \(\Theta(n^{2i})\), \(S(n) \in \Theta(n^{2i})\).

(d) Since \((1+x^2)^{-1}\) is decreasing, 
\[
\int_1^{n+1} dx (1+x^2)^{-1} \leq S(n) \leq \int_0^n dx (1+x^2)^{-1}.
\]
Computing the integrals, 
\[
\arctan(n+1) - \frac{\pi}{4} \leq S(n) \leq \arctan(n).
\]
Since the lower and upper bounds are in \(\Theta(1)\), \(S(n) \in \Theta(1)\).

(e) Since \(\sqrt{i/1+x^2}\) is decreasing, 
\[
\int_1^{n+1} dx x/(1+x^2) \leq S(n) \leq \int_0^n dx x/(1+x^2).
\]
Computing the integrals, 
\[
\frac{1}{4} \ln \frac{1}{4}(1 + (n+1)^2) \leq S(n) \leq \frac{1}{4} \ln(1 + n^2).
\]
Since the lower and upper bounds are in \(\Theta(\log n)\), \(S(n) \in \Theta(\log n)\).

(f) Since \(\sqrt{2^i}\) is increasing, 
\[
\int_0^n dx 2^{i^2} \leq S(n) \leq \int_1^{n+1} dx 2^{i^2}.
\]
Computing the integrals, 
\[
\frac{1}{2}\ln(2^n - 1) \leq S(n) \leq \frac{1}{2}\ln(2^{n+1} - 2).
\]
This time, the lower bound is in \(\Theta(2^n)\) and the upper bound is in \(\Theta(2^{n+1} - 2)\). The lower and upper bounds do not have the same asymptotic behavior, \(2^{n^2} \in o(2^{n^2+2n})\). Therefore we cannot immediately extract the \(\Theta\)-behavior for \(S(n)\).

The bounds provided by the integration method are too loose. We can get tighter bounds with a simpler analysis. The largest term in the sum is \(n2^{n^2}\) and there are \(n\) terms, so
\[
n2^{n^2} \leq S(n) \leq n^22^{n^2}.
\]
The lower bound is asymptotically tight. By the result above, \(S(n-1) \leq (n-1)2(2(n-1)^2)\) and \(S(n) = S(n-1) + n2^{n^2}\), therefore
\[
S(n) \leq n2^{n^2} + (n-1)2^{n(n-1)^2} = n2^{n^2} \left(1 + 2 \left(\frac{2(n-1)^2}{n} - 2\right)\right) \leq n2^{n^2} \left(1 + 2n2^{-2n}\right) \leq \frac{3}{2}n2^{n^2},
\]
(because \(x^2-x \leq \frac{1}{2}\) for \(x \geq 0\)). Therefore, \(S(n) \in \Theta(n2^{n^2})\), because \(n2^{n^2} \leq S(n) \leq \frac{3}{2}n2^{n^2}\).

**Exercise 9.9.**

(a) This is just the sum written out.

(b) Multiply the expression in (a) by 2 on both sides.

(c) Subtract (a) from (b): \(2S(n) - S(n) = -2^1 - 2^2 - 2^3 - \cdots - 2^n + n2^{n+1} = n2^{n+1} - \sum_{i=1}^n 2^i\).

(d) Use (c) with \(n \to 2n\), \(S(n) = n2^{n+1} - 2(2^n - 1) = (n-1)2^{n+1} + 2\).

**Exercise 9.10.** The red areas in the figure are larger than the green ones because the slope of \(\ln x\) is decreasing, \((\ln x)' = -1/x^2 < 0\). Computing the integral replaces each red region in the rectangle with the corresponding green region, therefore we get a lower bound,
\[
\int_{3/2}^n dx \ln x \leq \sum_{i=2}^n \ln i = \ln n!.
\]
Evaluating the integral on the left, we get
\[
\left(n + \frac{1}{2}\right) \ln \left(n + \frac{1}{2}\right) - \left(n + \frac{1}{2}\right) - \frac{1}{2} \ln \frac{3}{2} + \frac{1}{2} \leq \ln n!.
\]
Exponentiate both sides to get
\[
n! \geq \left(n + \frac{1}{2}\right)^{n+\frac{1}{2}} e^{-n} e \left(\frac{3}{2}\right)^{3/2} = n^n e^{-n\sqrt{n}} e \left(\frac{3}{2}\right)^{3/2} \left(n + \frac{1}{2}\right)^{n+1/2}.
\]
Lastly, using the approximation \((1 + \frac{1}{x})^x \approx e^x\),
\[
\left(n + \frac{1}{2}\right)^{n+\frac{1}{2}} = \left[\left(1 + \frac{1}{n+1}\right)^{2n}\right]^{(n+\frac{1}{2})/2n} \approx e^{\frac{1}{2} + \frac{1}{n+1}} = \sqrt{e} \cdot e^{1/4n}.
\]
And since \(e^{1/4n} = 1 + \Theta(1/4n)\), we get the desired approximation \(n! \approx n^n e^{-n} \sqrt{n}(2e/3)^{3/2}\).
Chapter 10

Pop Quiz 10.1. \(27 = 3 \times 7 + 6 \ (r = 6)\). By setting \(q = 3, 2, 1, 0, -1, \ldots\) we get \(r = 6, 13, 20, 27, \ldots\). The smallest positive remainder is 6.

Exercise 10.2. The proofs all use Definition 10.2.  
(a) \(0 = 0 \cdot d \ (q = 0)\), so \(d(0).\)  
(b) Suppose \(d|m\) and \(d'|n\), so \(m = qd\) and \(n = q'd'. \) Then \(mn = (qq')dd'\). That is \(dd'|mn\) (quotient = \(qq').\)  
(c) Suppose \(d|m\) and \(m|n\), so \(m = qd\) and \(n = q'm. \) Then, \(n = qqd\) so \(d|n\) (quotient = \(q'q).\)  
(d) Suppose \(d|n\) and \(d|m, \) so \(n = qd\) and \(m = q'd. \) Then \(n + m = (q + q')d. \) That is \(d|n + m\) (quotient = \(q + q').\)  
(e) Suppose \(d|n,\) so \(n = qd. \) For \(x \in \mathbb{N}, \) \(xn = qxd,\) so \(xd|xn\) (quotient = \(q).\)  
(f) Suppose \(d|m+n\) and \(d|m, \) so \(m+n = qd\) and \(m = q'd. \) Then, \(n = qd-n = qd-q'd = (q-q')d. \) That is \(d|n\) (quotient = \(q - q').\)

Exercise 10.3.  
(a) Let \(P(n) = n\) is divisible by a prime. \(P(2)\) is \(T\) because 2 is a prime. Suppose \(P(2), \ldots, P(n)\) are all \(T. \) We show that \(P(n + 1)\) is \(T. \) If \(n + 1\) is prime, then \(n + 1\) is divisible by the prime \(n + 1. \) Otherwise, \(n + 1\) is composite: \(n + 1 = k\ell, \) where \(2 \leq \ell \leq n. \) By the induction hypothesis, \(\ell\) is divisible by a prime, so \(\ell = qp\) where \(p\) is prime. Therefore \(n + 1 = kqp\) which shows that \(n + 1\) is also divisible by the prime \(p. \) By induction, \(P(n)\) is \(T\) for all \(n \geq 2.\)

(b) Suppose there are finitely many primes. Then, there is a largest prime \(p. \) Consider \(p! + 1,\) which has a remainder of 1 when divided by \(2, 3, \ldots, p. \) By part (a), \(p! + 1\) must be divisible by a prime. This prime must therefore be larger than \(p\) contradicting \(p\) being the largest prime. Therefore there are infinitely many primes.

Pop Quiz 10.4. \(\gcd(n, 0) = n\) because \(n|0\) and \(n|n.\)  
\(\gcd(n, n) = n\) because \(n|n.\)  
\(\gcd(n, 1) = 1\) because the largest divisor of 1 is 1.  
\(\gcd(n, p) = \begin{cases} 1 & \text{if } p \text{ does not divide } n; \\ p & \text{if } p \text{ does divides } n. \end{cases}\) (Because the only divisors of \(p\) are 1 and \(p.\))

Exercise 10.5.  
\begin{align*}  
gcd(34, 55) &= gcd(21, 34) = 21 = 55 - 34 \\ &= gcd(13, 21) = 13 = 34 - 21 = 34 - (55 - 34) = 2 \cdot 34 - 55 \\ &= gcd(8, 13) = 8 = 21 - 13 = (55 - 34) - (2 \cdot 34 - 55) = 2 \cdot 55 - 3 \cdot 34 \\ &= gcd(5, 8) = 5 = 13 - 8 = (2 \cdot 34 - 55) - (2 \cdot 55 - 3 \cdot 34) = 5 \cdot 34 - 3 \cdot 55 \\ &= gcd(3, 5) = 3 = 8 - 5 = (2 \cdot 55 - 3 \cdot 34) - (5 \cdot 34 - 3 \cdot 55) = 5 \cdot 55 - 8 \cdot 55 \\ &= gcd(2, 3) = 2 = 5 - 3 = (5 \cdot 34 - 3 \cdot 55) - (5 \cdot 55 - 8 \cdot 55) = 13 \cdot 34 - 8 \cdot 55 \\ &= gcd(1, 2) = 1 = 3 - 2 = (5 \cdot 55 - 8 \cdot 55) - (13 \cdot 34 - 8 \cdot 55) = 13 \cdot 55 - 21 \cdot 34 \\ &= gcd(0, 1) = 1 \end{align*}

On the right we show how to write each remainder in the GCD algorithm as a linear combination of the original two numbers. The final result is 
\(\gcd(34, 55) = 1 = 13 \times 55 - 21 \times 34.\)

Exercise 10.6. \(x = 1, y = 1\) gives \(mx + ny = 21;\) \(x = -2, y = 1\) gives \(mx + ny = 3;\) You cannot get a linear combination that is smaller than \(3\) and \(\gcd(6, 15) = 3.\)

Exercise 10.7.  
(a) Suppose \(d|m,\) By Bezout’s identity, there are \(x, y\) for which \(\gcd(m, d) = mx + dy.\) Multiply both sides by \(n\) to get 
\(\gcd(m, d) \cdot n = xmn + ynd.\) 
\(d\) divides the second term on the right (it is a multiple of \(d).\) \(d\) also divides the first term on the right because \(d|m,\) so \(mn = ad.\) Therefore, the RHS is \((xa + yd)n,\) that is \(d\) divides the sum on the RHS. Therefore \(d\) must divide the LHS which is what was to be shown.
(b) We are given that \( \gcd(d, d') = 1 = dx + d'y \) (Bezout’s identity). Multiply both sides of Bezout’s identity by \( n \) to get

\[
n = xdn + yd'n.
\]

Since \( d|n, n = ad \); since \( d'|n, n = a'd' \). Rewriting the equation above,

\[
n = xa'd'd' + yadd' = (xa' + ya)d'd',
\]

which means \( dd'|n \) as was to be shown.

(c) Let \( D = \gcd(m, \ell) \) and \( D' = \gcd(n, \ell) \). By Bezout’s identity, \( D = mx + \ell y \) and \( D' = nx' + \ell y' \).

Multiplying these two equations,

\[
DD' = (mx + \ell y)(nx' + \ell y') = mn(xx') + \ell(ynx' + mxy' + \ell yy').
\]

Since \( DD' > 0 \), the RHS is a positive linear combination of \( mn \) and \( \ell \). The smallest positive linear combination of \( mn \) and \( \ell \) is \( \gcd(mn, \ell) \), so \( \gcd(mn, \ell) \leq DD' \).

To show the reverse inequality, that \( DD' \leq \gcd(mn, \ell) \), it suffices to show that \( DD' \) is a common divisor of \( mn \) and \( \ell \) because then it must be at most the greatest common divisor. By Exercise 10.2(b), \( DD'|mn \). We show that \( DD'|\ell \). To do so, we will use part (a) of this exercise. For part (a) to apply, we must show that \( \gcd(D, D') = 1 \). Note that \( \gcd(D, D')|m \)

because \( \gcd(D, D')|D \) and \( D|m \); similarly, \( \gcd(D, D')|n \). Therefore, \( \gcd(D, D') \) is a common divisor of \( m \) and \( n \), which implies

\[
\gcd(D, D') \leq \gcd(m, n) = 1.
\]

Thus, \( \gcd(D, D') = 1 \). By part (a), since \( D|\ell \) and \( D'|\ell \) (why?), it follows that \( DD'|\ell \). Therefore \( DD' \) is a common divisor of \( mn \) and \( \ell \) and hence \( DD' \leq \gcd(mn, \ell) \).

Since \( \gcd(mn, \ell) \leq DD' \) and \( DD' \leq \gcd(mn, \ell) \), it follows that \( \gcd(mn, \ell) = DD' \).

[Note: It is essential that \( \gcd(m, n) = 1 \) (consider \( m = n = \ell = 5 \).]

(d) Using the same notation in (b), we are to show that \( \gcd(mn, \ell) = 1 \) if and only if \( D = 1 \) and \( D' = 1 \).

First, suppose \( D = 1 \) and \( D' = 1 \). In (b), (\( * \)) showed that \( \gcd(mn, \ell) \leq DD' = 1 \), which proves \( \gcd(mn, \ell) = 1 \).

Now, suppose \( \gcd(mn, \ell) = 1 \). Any divisor of \( mn \) and \( \ell \) is also a divisor of \( mn \) and \( \ell \) so \( \gcd(m, \ell) \leq \gcd(mn, \ell) = 1 \). Similarly, \( \gcd(n, \ell) \leq \gcd(mn, \ell) = 1 \). We conclude that \( \gcd(m, \ell) = \gcd(n, \ell) = 1 \).

(e) Let \( D = \gcd(\gcd(\ell, m), n) \) and \( D' = \gcd(\ell, \gcd(m, n)) \). Since \( D|gcd(\ell, m), d|\ell \) and \( d|m \), and also \( D|n \). Since \( D|m \) and \( D|n \), by GCD fact (ii) on page 129, \( D|\gcd(m, n) \). Therefore \( D \) is a common divisor of \( \ell \) and \( \gcd(m, n) \). This means \( D \leq D' \).

A similar argument proves reversed inequality \( D' \leq D \): \( D' \) divides \( \ell \), \( m \) and \( n \); this means \( D'|\gcd(\ell, m) \) and hence \( D' \) is a common divisor of \( \gcd(\ell, m) \) and \( n \). It follows that \( D' \leq D \).

Therefore, \( D = D' \).

**Exercise 10.8.** We prove the claim by induction on \( n \). The base case is \( n = 2 \): if \( p | q_1 q_2 \), then, by Euclid’s Lemma, \( p | q_1 \) or \( p | q_2 \). For the induction step, suppose that for any \( n \) primes, if \( p | q_1 \cdots q_2 \) then \( p \) equals one of the \( q_i \). Consider any \( n + 1 \) and suppose that \( p | q_1 \cdots q_n q_{n+1} \). That is \( p | (q_1 \cdots q_n) q_{n+1} \). By Euclid’s Lemma, either \( p | q_{n+1} \) or \( p | q_1 \cdots q_n \). In the former case, because \( q_{n+1} \) is prime, \( p = q_{n+1} \); in the latter case, by the induction hypothesis \( p \) equals one of the \( q_i \). In either case \( p \) equals one of the \( n + 1 \) primes \( q_1, \ldots, q_{n+1} \), proving the claim for \( n + 1 \). The claim now follows by induction for all \( n \geq 2 \).

**Pop Quiz 10.9.** This is just the Fundamental Theorem of Arithmetic in disguise. By the fundamental theorem of arithmetic, every \( n \geq 2 \) is a product of primes, \( n = p_1 p_2 \cdots p_a \); \( a_1 \) is the number of times \( 2 \) appears in this product; \( a_2 \) is the number of times \( 3 \) appears in this product; and so on. The \( a_i \) must be unique for every \( n \) because if it is not unique for some \( n \), then that \( n \) is a product of primes in two different ways, which cannot be.

**Exercise 10.10.** We must show \( \gcd(kM_1, kM_2) = k \). The only divisors of \( kM_1 \) are 1, \( k \) and \( M_1 \), since \( k \) and \( M_1 \) are different primes Similarly, the only divisors of \( kM_2 \) are 1, \( k \) and \( M_2 \). The largest that is common to these two sets is \( k \).

**Exercise 10.11.** We have: \( a \equiv b \pmod{d} \) and \( r \equiv s \pmod{d} \). That is,

\[
a - b = k_1 d \quad \text{and} \quad r - s = k_2 d.
\]

458
Solutions to Quizzes & Exercises

30. Pop Quiz 10.12. We observe that $5^2 \equiv 1 \pmod{3}$. Therefore by Exercise 10.10(c), $5^{2n} \equiv 1^{2n} \pmod{3}$. Note $1^{2n} = 1$. So, $5^{2014} \equiv 1 \pmod{3}$. Since $5 \equiv 2 \pmod{3}$, using Exercise 10.10(a), $5^{2014} \equiv 2 \pmod{3}$. The remainder is 2.

Exercise 10.14. We use a proof by contradiction. Suppose 15 does have a multiplicative inverse $k$. Then $15k \equiv 1 \pmod{6}$, that is $15k - 1 = 6ph$, or $15k - 6 = 1 \equiv 3$ divides the LHS but not the RHS, a contradiction. So, 15 does not have a multiplicative inverse.


(a) (i) If $k = 0$ then $0^p = 0$ and $0 - 0 = 0$ is divisible by $p$. If $k = p$ then $p^p - p = p(p^{p-1} - 1)$ is a multiple of $p$ and so is divisible by $p$.

(ii) If $i \in \{1, \ldots, p-1\}$, then $\gcd(p, i) = 1$. If $p|ik$ then by Exercise 10.7(a), $p|k$, that is $k$ is a multiple of $p$, a contradiction. So, $p$ does not divide $ik$ and $ik$ is not a multiple of $p$.

(iii) Immediate from Theorem 10.10 on page 133, because $\gcd(k, p) = 1$.

(iv) Since $ik$ is not a multiple of $p$, by part (ii), $\alpha_i \neq 0$. Thus, $\alpha_i \in \{1, 2, \ldots, p-1\}$.

Suppose $i, j \in \{1, \ldots, p-1\}$. We show, by contradiction, that if $i \neq j$ then $\alpha_i \neq \alpha_j$.

Suppose $\alpha_i \neq \alpha_j$. Then $ik = q_i + \alpha_i$ and $jk = q_j + \alpha_j$. Subtracting $ik - jk = (q_i - q_j)p$. That is, $ik \equiv jk \pmod{p}$. By (iii), $i \equiv j \pmod{p}$. Since $i, j \in \{1, 2, \ldots, p-1\}$, this means $i = j$, a contradiction. Thus, $\alpha_i \neq \alpha_j$. Since no two $\alpha_i$ are equal, $\alpha_1, \alpha_2, \ldots, \alpha_{p-1}$ is a permutation of $1, 2, \ldots, p-1$, which means $\prod_{i=1}^{p-1} \alpha_i = (p-1)!$.

(v) Since $ik \equiv \alpha_1 \pmod{p}$, by repeated use of Exercise 10.10, $\prod_{i=1}^{p-1} ki \equiv \prod_{i=1}^{p-1} \alpha_i \pmod{p}$. $\prod_{i=1}^{p-1} ki = k^{p-1}(p-1)!$ and by (iv), $\prod_{i=1}^{p-1} \alpha_i = (p-1)!$, therefore $k^{p-1}(p-1)! \equiv (p-1)! \pmod{p}$. ($\ast$)

If $p$ divides $(p-1)!$, by Lemma 10.8 on page 131 (Euclid’s Lemma), $p$ divides one of the terms in the product. Since every term in the product is less than $p$, that is not possible, so $p$ does not divide $(p-1)!$. The only other divisor of $p$ is 1, so $\gcd(p, (p-1)!)$ = 1. By Theorem 10.10 on page 133, we can cancel the $(p-1)!$ from both sides of ($\ast$) to get $k^{p-1} \equiv 1 \pmod{p}$.

Multiplying both sides by $k$, since $k \equiv k \pmod{p}$, gives Fermat’s Little Theorem.

(b) When $k$ is a multiple of $p$, there is no multiplicative inverse since $\gcd(k, p) = p > 1$. When $k$ is not a multiple of $p$, we just proved above that $k^{p-1} \equiv 1 \pmod{p}$. Let $k^{-1} \equiv k^{p-2} \pmod{p}$. Then $k \cdot k^{-1} \equiv k^{p-1} \equiv 1 \pmod{p}$. That is, $k^{p-2}$ is the multiplicative inverse of $k$.

(c) (i) We find $x, y$ such that $8x + 19y = 1$ using the remainders in Euclid’s GCD-algorithm:

$\gcd(8, 19) = \gcd(3, 8)$

$= \gcd(2, 3)$

$= \gcd(1, 2) = 1$

rem(19, 8) = 3 = \begin{align*}
-8 \cdot 2 + 19 \\
-8 \cdot 3 + 2 \\
-8 \cdot 5 + 19 \\
-8 \cdot 6 + 19 \\
-8 \cdot 7 + 19
\end{align*}$

rem(8, 3) = 2 = \begin{align*}
8 - (\cdot 2 + 19) \cdot 2 \\
8 - 5 \cdot 19 \\
8 - 6 \cdot 19 \\
8 - 7 \cdot 19
\end{align*}$

rem(3, 2) = 1 = \begin{align*}
3 - 2 \\
3 - 5 \cdot 19 \\
3 - 6 \cdot 19 \\
3 - 7 \cdot 19
\end{align*}$

Therefore $x = -7$ and $8^{-1} = \text{rem}(-7, 19) = 12$ (for modulus 19). You can verify that $8 \times 12 \equiv 1 \pmod{19}$ because $8 \times 12 - 1 = 19 \times 5$.

(ii) $8^{-1} \equiv 8^{17} \pmod{19}$. We observe that $8^3 \equiv -1 \pmod{19}$. Therefore, $8^{15} \equiv (-1)^5 \equiv -1 \pmod{19}$.

Finally, $8^{17} \equiv -64 \equiv 12 \pmod{19}$, so $8^{-1} = 12$ (for modulus 19).
Exercise 10.15. \[ M^{p-1} M_a \equiv M^{\ell_1} k \pmod{p} \]. Assuming \( M \) is not a multiple of \( p \), by Fermat’s Little Theorem, \( M^{p-1} \equiv 1 \pmod{p} \). Multiplying both sides by \( k \) gives \( M^{p-1} k \equiv k \pmod{p} \), that is,
\[ M^{p-1} M_a \equiv M^{\ell_1} k \equiv k \pmod{p}. \]
So Charlie obtains \( k \) by computing \( \text{rem}(M^{p-1} M_a, p) \).

Exercise 10.16. Alice encrypts to \( M_a \equiv M^{225} \pmod{391} \) and Bob decrypts with \( M^{27} \pmod{391} \).
For example, with \( M = 2 \), we have
\[ 2^7 \equiv 128 \rightarrow 2^{14} \equiv 128^2 \equiv 353 \rightarrow 2^{28} \equiv 353^2 \equiv 271 \rightarrow 2^{56} \equiv 271^2 \equiv 324 \rightarrow 2^{112} \equiv 324^2 \equiv 188, \]
and finally we have \( 2^{225} \equiv 2 \cdot 188^2 \equiv 308 \) (mod 391). Bob decrypts as follows:
\[ 308^3 \equiv 246 \rightarrow 308^6 \equiv 246^2 \equiv 302 \rightarrow 308^{12} \equiv 302^2 \equiv 101 \rightarrow 308^{24} \equiv 101^2 \equiv 35 \rightarrow 308^{48} \equiv 35^2 \equiv 52, \]
and finally we have \( 308^{97} \equiv 308 \cdot 52^2 \equiv 2 \pmod{391} \). Here is the table of results for \( M = 2, \ldots, 10, \)

<table>
<thead>
<tr>
<th>( M )</th>
<th>( M_a )</th>
<th>( M_a )</th>
<th>( M_a )</th>
<th>( M_a )</th>
<th>( M_a )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
</tr>
</tbody>
</table>

Bob’s decryption
\[ 2 \quad 3 \quad 4 \quad 5 \quad 6 \quad 7 \quad 8 \quad 9 \quad 10 \]
(Bob always recovers \( M \).)

Exercise 10.17. Let \( n = pq; M_a \equiv M^e \pmod{n} \). We decode using \( M^d \equiv M^{ed} \pmod{n} \), and must show \( M^{ed} \equiv M \) (mod \( n \)) (i.e. we recover the correct message for any \( M \)).
(a) Since \( ed \equiv 1 \pmod{(p-1)(q-1)} \), this means that \( ed - 1 \) is divisible by \( (p-1)(q-1) \), or there is some \( k \) for which \( ed - 1 = k(p-1)(q-1) \).
(b) (i) By (a) \( ed - 1 = k(p-1)(q-1) \), so \( M^{ed-1} \equiv M^{k(p-1)(q-1)} \).
   (ii) By Fermat’s Little Theorem, \( M^{p-1} \equiv 1 \pmod{p} \) (because \( p \) does not divide \( M \)). This means \( M^{p-1} - 1 = \alpha \) for an integer \( \alpha \), or that \( M = 1 + \alpha \). Therefore,
   \[ M^{ed-1} = M^{k(p-1)(q-1)} = (M^{p-1})^{k(q-1)} = (1 + \alpha)^{k(q-1)}. \]
(iii) By the Binomial Theorem,
   \[ (1 + \alpha)^{k(q-1)} = 1 + \sum_{i=1}^{k(q-1)} \binom{k(q-1)}{i} \alpha^i = 1 + \sum_{i=1}^{k(q-1)} \binom{k(q-1)}{i} \alpha^i p^{i-1}. \]
   We could also use \( \alpha^r - 1 = (a - b)(1 + a + a^2 + \cdots + a^{r-1}) \) with \( a = 1 + \alpha \):
   \[ (1 + \alpha)^{k(q-1)} - 1 = \sum_{i=1}^{k(q-1)} \binom{k(q-1)}{i} \alpha^i p^{i-1}. \]
   Either way, \( \beta \) is a sum of integers, hence \( \beta \) is an integer. We have proved:
   \[ M^{ed-1} = (1 + \alpha)^{k(q-1)} = 1 + \beta p. \]
   (iv) From (iii), \( M^{ed-1} - 1 = \beta p \), that is \( p|M^{ed-1} - 1 \).
(c) By (b), either \( p \) divides \( M \) or if not then it must divide \( M^{ed-1} - 1 \). That is \( p \) must divide their product \( M^{ed} - M \). Everything is symmetric with respect to \( p \) and \( q \) and so using exactly the same reasoning, \( q|M^{ed}-M \).
(d) Since \( \gcd(p, q) = 1 \) and both \( p|M^{ed} - M \) and \( q|M^{ed} - M \), we can apply Exercise 10.7(b) on page 130 to obtain \( pq|M^{ed} - M \).
(e) From (d), by definition, \( M^{ed} \equiv M \pmod{pq} \). Bob can decode by computing \( \text{rem}(M^d, pq) \).

Chapter 11

Pop Quiz 11.1.
(a) \( V = \{1, 2, 3, 4, 5, 6\}; \)
   \[ E = \{(1, 2), (2, 3), (3, 4), (1, 4)\}; \]
(b) \( V = \{a, b, c, d, e, f, g\}; \)
   \[ E = \{(a, b), (b, c), (c, d), (a, d)\}; \]
(c) \( V = \{i, j, k, l, m, n\}; \)
   \[ E = \{(i, j), (j, k), (i, m), (j, n), (k, m), (k, n)\}; \]
(d) \( V = \{i, j, k, l, m, n\}; \)
   \[ E = \{(i, j), (i, k), (i, l), (j, k), (j, l), (m, n)\}; \]
Exercise 11.2. Isomorphic graphs: {I, II}

(a) Since you only relabeled vertices, the number of vertices did not change; every edge still exists with relabeled end points, so the number of edges did not change.

(b) In the relabeling, suppose vertex \( v \) is relabeled to \( \ell(v) \). Then every edge in the graph \( (v, w) \) becomes \( (\ell(v), \ell(w)) \) and \( \deg(v) \) becomes \( \deg(\ell(v)) \). Every vertex \( w \) which contributes to \( \deg(v) \) is relabeled to a vertex \( \ell(w) \) which contributes to the degree of \( \ell(v) \). Therefore, the degree of each vertex does not change.

(c) Suppose \( v_1v_2 \cdots v_k \) is a path between. Every edge \( (v_i, v_{i+1}) \) exists in the graph since \( v_1v_2 \cdots v_k \) is a path. Therefore, from the relabeling, edge \( (\ell(v_1), \ell(v_{i+1})) \) is present in the relabeled graph. Hence \( \ell(v_1)\ell(v_2) \cdots \ell(v_k) \) is a path in the relabeled graph.

(d) Every path in the graph between any two nodes is preserved as a relabeled path between the relabeled two vertices. This includes shortest paths.

Exercise 11.3. There were some trick questions in this exercise.

(a) An isomorphism preserves all paths (see Exercise 11.2). In the first graph, there is a path between every pair of vertices, but not so in the second, so the graphs cannot be isomorphic.

(b) This is a trick question. All graphs with the degree sequence [3,3,2,1,1] are isomorphic. To see this label the vertices \( A, B, C, D, E \) (highest to lowest degree).

Since \( E \) cannot have any more neighbors, and \( B \) must have two more neighbors, it must be that \( B \) is connected to the other two vertices, completing the picture as shown on the right. There is no other way to construct a graph with this degree sequence. (Not all degree sequences can be realized by different, non-isomorphic, graphs. Another classic example is \([n-1,1,1,\ldots,1]\).)

(c) Another trick question. There is no graph with this degree sequence. Here’s a hint as to why: the sum of the degrees is 13 (more on this later).

Pop Quiz 11.4. This graph cannot exist because there are an odd number of odd-degree vertices.

Exercise 11.5.

(a) If every degree is positive, \( 2m = \sum \delta_i \geq n \), so \( m \geq n/2 \). Example: \( \circ \circ \circ \circ \circ \).

(b) Equivalently, we compute the maximum number of edges a graph with a degree 0 vertex can have. Let \( v \) be the degree 0 vertex and \( n \) the number of vertices. Every vertex other than \( v \) can have an edge to every vertex other than \( v \), so ever vertex other than \( v \) has degree \( n-1 \). The number of edges is \( \frac{1}{2}(n-1)(n-2) \) (half the sum of the degrees). This is the maximum number of edges for a graph with a degree 0 vertex. So, if a graph has \( 1 + \frac{1}{2}(n-1)(n-2) \) edges, it cannot have a degree 0 vertex.

(c) The sum of the degrees is \( 5 \times 3 = 15 \). There are no such graphs, since the sum of the degrees must be even. You could also argue that such a graph would have 5 vertices of odd degree, which violates Corollary 11.3.

Exercise 11.6.

(a) The dotted edge creates a cycle, and a tree is connected with no cycles.

(b) Steps 1–5 are not connected. After step 6, any new edge would create a cycle.

(c) This is an important result so we give two different proofs.

Theorem 30.1. A graph with fewer than \( n - 1 \) edges is not connected.

Proof. We prove the claim by induction on \( n \). The base case is \( n = 2 \) in which case the graph with 0 edges is clearly not connected. Consider any graph with \( n + 1 \) vertices and fewer than \( n \) edges. Every vertex cannot have degree at least 2 (the sum of the degrees would be at least \( 2(n+1) \) implying at least \( n+1 \) edges), so some vertex \( v \) has degree less than 2. If \( \deg(v) = 0 \)
then the graph is disconnected as was to be shown.

we show the situation with \( \deg(v) = 1 \) on the right. The shaded region is the rest of the graph, other than \( v \), and there is an edge \( e \) from \( v \) to one node in the shaded region. Remove \( v \) and \( e \) from the graph. The shaded graph that remains has \( n \) vertices and fewer than \( n - 1 \) edges (we removed one vertex and edge). By the induction hypothesis, this residual (shaded) graph is not connected: two vertices (illustrated by \( x \) and \( y \)) are not connected by any path. Adding back \( v \) and \( e \) cannot create a path between \( x \) and \( y \); so \( x \) and \( y \) remain disconnected in the original graph. We have proved that any graph with \( n + 1 \) vertices and fewer than \( n \) edges is not connected, so the theorem follows by induction.

We now give a proof of a more general result from which Theorem 30.1 follows. A component in a graph is a “maximal” set of vertices that is connected. The component of a vertex \( v \) is all the vertices connected to \( v \),

\[
C(v) = \{ u \mid u \text{ is connected to } v \text{ by a path} \}.
\]

Any vertex in a component can be used to define that component, that is if \( u \in C(v) \) then \( C(u) = C(v) \); any vertex in \( C(v) \) is connected to \( u \) by going first from \( u \) to \( v \) and then to the vertex, hence \( C(v) \subseteq C(u) \); similarly any vertex in \( C(u) \) is connected to \( v \) by going first from \( v \) to \( u \) and then to the vertex, hence \( C(u) \subseteq C(v) \).

\[
\begin{array}{ccc}
1 \text{ component} & 2 \text{ components} & 6 \text{ components} \\
\end{array}
\]

A graph is connected if it has only one component. A graph with \( n \) isolated vertices (and no edges) has \( n \) components. If you add an edge between two vertices in the same component, you do not change any of the components. If you add an edge between two vertices in different components, you merge those two components, decreasing the number of components by 1.

**Lemma 30.2.** Adding an edge can decrease the number of components by at most 1.

We use this lemma to prove our general result:

**Theorem 30.3.** A graph with \( n \) vertices and \( e \) edges has at least \( n - e \) components.

**Proof.** Start with the \( n \) isolated vertices and no edges (\( n \) components) and add the edges one by one, each time decreasing the number of components by at most one. So the number of components decreases by at most \( e \), leaving at least \( n - e \) components. The formal proof would be by induction on \( e \), the number of edges in the graph. In the induction step you start with \( e + 1 \) edges and remove an edge. By the induction hypothesis there will be at least \( n + 1 - e \) edges. Now add back the edge and apply Lemma 30.2 to conclude there are at least \( n - e \) components. □

Theorem 30.3 implies Theorem 30.1 because if \( e < n - 1 \), then

\[
\text{number of components} > n - (n - 1) = 1.
\]

This means the number of components is at least 2 and the graph is disconnected.

(d) Suppose the graph has \( n + k \) edges, where \( k \geq 0 \). We first prove the case that the graph has a single component. This means that for any set of vertices there is at least one edge from a vertex in the set to a vertex not in the set (otherwise that set of vertices is disconnected from the rest of the graph).

Let us build a connected component by adding one vertex at a time. Start at any vertex \( v_1 \). There must be an edge from \( v_1 \) to a second vertex \( v_2 \). So we have built the set \( v_1, v_2 \). After we have built the set \( v_1, v_2, \ldots, v_i \), there must be an edge from a vertex in our set to an \( i + 1 \)th vertex \( v_{i+1} \). Continue this process until we have built the set containing all the nodes \( v_1, \ldots, v_n \) using \( n - 1 \) edges. By construction, this set of vertices is connected using only the \( n - 1 \) edges — this set of \( n - 1 \) edges is called a *spanning tree*.

There is at least one more edge in the graph, say \((v_i, v_j)\), since the graph has at least \( n \) edges. Before adding the edge \((v_i, v_j)\), there was a path from \( v_i \) to \( v_j \) in the spanning tree. The edge \((v_i, v_j)\) together with this path is a cycle.
The formal proof of this claim is by strong induction. In the induction step, take any graph with \( n \) vertices and \( n + k \) edges. Remove an edge \( e \). If the graph remains connected, you can show that adding back \( e \) creates a cycle. If \( e \) is disconnected then you can one of the components has at least as many edges as vertices and hence contains a cycle. Adding back \( e \) won’t remove that cycle.

Now consider a graph with more than one component. Suppose the graph has \( \ell \) components with \( n_1, n_2, \ldots, n_\ell \) vertices in each component, and \( e_1, e_2, \ldots, e_\ell \) edges in each component: \( n_1 + n_2 + \cdots + n_\ell = n \) and \( e_1 + e_2 + \cdots + e_\ell = e \). We claim that for some \( i^* \), \( e_{i^*} \geq n_{i^*} \) because if not, then \( e_i < n_i \) for every \( i \) and

\[
e = e_1 + e_2 + \cdots + e_\ell < n_1 + n_2 + \cdots + n_\ell = n,
\]

which cannot be since \( e \geq n \). There must therefore be a cycle in the \( i^* \)th component and hence in the graph.

(e) Since this is an “if and only if”, there are two parts to the proof.

1. Suppose \( G \) is a connected graph with \( n \) vertices that is a tree. By (d), if \( G \) has fewer than \( n - 1 \) edges, then it cannot be connected, so \( G \) has at least \( n - 1 \) edges. If \( G \) has \( n \) or more edges, then by (e) it has a cycle and therefore is not a tree. Hence \( G \) must have \( n - 1 \) vertices.

2. Suppose \( G \) is a connected graph with \( n \) nodes and \( n - 1 \) edges. We must show that \( G \) is a tree. We need to show that there are no cycles. Suppose \( G \) contains a cycle. Remove an edge on this cycle. Every vertex remains connected with every other vertex, hence the graph remains connected but has \( n - 2 \) edges. This contradicts (d). Hence \( G \) has no cycles.

Exercise 11.7.

(a) The faces become regions (polygons). One of the faces becomes the external (unbounded) region.

(b) There is only one external face, so \( F = 1 \). For a tree, we know from Exercise 11.6 that \( E = V - 1 \). Therefore,

\[
F + V - E = 1 + V - (V - 1) = 2.
\]

(e) Consider a connected graph that is not a tree (i.e. has cycles).

(i) Look at what happens when you remove one edge from a cycle.
There are two cases: the edge is between two internal faces (left) and the internal faces merge into one internal face; the edge is between an internal face and the external face (right) and the external face merges with the internal face to make a larger external face. In either case, the number of faces $F$ decreases by 1. We removed one edge but the number of vertices remained the same. So,

$$\Delta E = -1; \quad \Delta F = -1; \quad \text{and} \quad \Delta V = 0.$$  

(ii) Paths that do not use removed edge are not affected by removing an edge. Paths that use the removed edge can go the other way around the cycle (instead of using the edge). Thus, if there was a path between two vertices, there still will be one.

(iii) Removing an edge from a cycle does not affect connectivity.

(iv) When you are done removing edges from cycles, there are no more cycles, and the graph is connected. Therefore it is a tree. In this process, $F \to F + \Delta F$, $E \to E + \Delta E$ and $V \to V + \Delta V$. For a tree, we proved in (d) that (faces) + (vertices) − (edges) = 2, therefore

$$F + \Delta F + V + \Delta V - (E + \Delta E) = 2.$$  

Since $\Delta V = 0$ and $\Delta F = \Delta E$, we get

$$F + V - E = 2,$$

where $F$, $V$ and $E$ are for the original graph with cycles.

(f) When you traverse around every face, $\sum f E(f)$ edges are traversed. Every edge is traversed twice: Edges on internal faces belongs to two faces and so are traversed once for each face; Edges on the external face that are not on an internal face are also traversed twice, back and forth. Since every edge is traversed twice, $\sum f E(f) = 2E$.

(g) Every internal face is bounded by at least 3 edges so $E(f) \geq 3$ for an internal face. If $V \geq 3$, the external face contains at least 3 vertices and hence $E$ (external face) $\geq 3$. Therefore $2E = \sum f E(f) \geq 3F$. That is $F \leq \frac{2}{3}E$. Using Euler’s Characteristic,

$$E = F + V - 2 \leq \frac{2}{3}E + V - 2 \rightarrow \frac{1}{3}E \leq V - 2 \rightarrow E \leq 3V - 6.$$  

In any planar graph with at least 3 vertices, $E \leq 3V - 6$.

In a planar graph, the number of edges is linear in the number of vertices. In $K_5$, $V = 5$, so $3V - 6 = 9$; yet, $E = 10$ which is greater, so $K_5$ cannot be planar.

(h) If there are no 3-cycles, then $E(f) \geq 4$ for every internal face $f$, because traversing around an internal face creates a cycle. The external face contains at least 3 vertices in which case going around the external face traverses at least 4 edges, so again $E(f) \geq 4$. Therefore, $2E = \sum f E(f) \geq 4F$ and, using Euler’s Characteristic,

$$E = F + V - 2 \leq \frac{1}{2}E + V - 2 \rightarrow \frac{1}{2}E \leq V - 2 \rightarrow E \leq 2V - 4.$$  

Any simple graph has no 2-cycles. In $K_{3,3}$ there are no 3-cycles because any path of odd length takes you from one side to the other. In $K_{3,3}$, $V = 6$, so $2V - 4 = 8$, but $E = 9$ which is larger, so $K_{3,3}$ cannot be planar.
Exercise 11.9. Vertices in Euler’s multi-graph are regions of Königsberg and edges are the bridges that connect two regions. Euler’s problem asks to start at some vertex in the graph, and follow a path of edges, ending at some other vertex. The requirement is that every edge must be traversed exactly once. Other than the start and end vertex, every other vertex, if entered using some (untraversed) edge must be exited using a different (untraversed) edge. This means every vertex, other than the start and end vertex, must have an even degree. Every vertex in Euler’s graph has an odd degree, so Euler’s problem is not solvable.

A path that Euler was seeking, which uses every edge is called an Euler tour. If the path starts and ends at the same vertex, it is called an Euler cycle.

Theorem 30.4. A connected graph has an Euler cycle if and only if every vertex has even degree. A connected graph has an Euler tour starting at $u$ and ending at $v$ if and only if the degrees of $u$ and $v$ are odd and every other vertex has even degree.

There are two parts to an if and only if proof. Try induction for the “hard part”.

Exercise 11.10. We highlight the fastest path on the right, which takes 11ms.

The fastest path is counterintuitive because it doesn’t always move “toward” the destination. When you take a course in algorithms you will learn how to systematically compute shortest paths when the edge weights are non-negative. The idea is to compute the shortest paths to all vertices simultaneously, starting with the closest vertex, then the next closest and so on. The technique is called dynamic programming and the algorithm is Dijkstra’s algorithm.

Pop Quiz 11.10.

(a) $V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$ and $E = \{(v_1 \to v_2), (v_3 \to v_1), (v_3 \to v_2), (v_2 \to v_4), (v_2 \to v_5), (v_3 \to v_4), (v_5 \to v_4), (v_6 \to v_7), (v_2 \to v_1), (v_4 \to v_2), (v_6 \to v_2)\}$.

(b) The graph is not (strongly) connected (there is no path from $v_7$ to any other node).

Exercise 11.11.

(a) $m, z, d$ are top-dogs.

(b) Let $t$ be a vertex with maximum out-degree (in case of ties, pick one arbitrarily). We prove that $t$ is a top-dog. We must show that $t$ dominates every other node $u$ (either $t$ beats $u$ or $t$ beats a vertex that beats $u$). Suppose, to the contrary, that there is some vertex $v$ which $t$ does not dominate. That is, $v$ beats $t$ and also beats everyone who $t$ beats. Then out-deg($v$) is at least $1 +$ out-deg($t$), which contradicts $t$ having maximum out-degree. Therefore such a $v$ does not exist.

(c) Let $v_1 \to v_2$ but $v_1 \to v_1$ for $i > 2$. So, $v_1$ wins one match. Let $v_2 \to v_1$ for $i > 2$. So, $v_2$ beats everyone but $v_1$. By construction, $v_1$ is a top-dog, having beaten just one vertex. Let $v_3 \to v_4 \to \cdots \to v_n \to v_3$, and the results of all other matches can be arbitrary. So, $v_2, \ldots, v_n$ all have out-degree at least 2. This means $v_1$ won the fewest possible matches and yet is a top-dog.

Chapter 12

Exercise 12.1.

(a) Let $|E|$ be the total number of partners that men have, which is the total number of partners women have since every partnership is between a man and a woman. Let $M$ be the number of men and $F$ the number of women. Then,

average partners per man $= \frac{|E|}{M} = \frac{|E|}{F}$ average partners per woman $\times \frac{F}{M}$.

Since $F/M = 50.8/49.2 \approx 1.0325$,

average partners per man $= \text{average partners per woman} \times 1.0325$.

Men have 3.25% more partners on average.
Exercise 12.5.

As for the boys, $Z$ is indifferent between the two matchings because he gets the same partner in both matchings. $X$ prefers $B$ to $C$ and $Y$ prefers $C$ to $B$ so the boys prefer the first matching.

Exercise 12.4.

A matching is stable if there is no pair of matches that is volatile.

$$
\begin{array}{ccc|ccc}
\text{X} & \text{Y} & \text{Z} & \text{A} & \text{B} & \text{C} \\
1. & \text{A} & \text{A} & \text{B} & \text{A} & \text{B} & \text{C} \\
2. & \text{B} & \text{C} & \text{A} & \text{Y} & \text{X} & \text{X} \\
3. & \text{C} & \text{B} & \text{C} & \text{X} & \text{Z} & \text{Y} \\
\end{array}
$$

The match $A-Z$ is “stable” because $Z$ is $A$’s top choice so $A$ will not wish to break her current match. The only possible volatile pair is $(X,Y)$ and $(B,C)$. Since $X$ prefers $B$ to $C$, this is not a volatile pair. Since there are no volatile pairs of matches, the matching is stable.

Again, $Z$ is $A$’s top choice, so the only possible volatile pair is $(X,Y)$ and $(B,C)$. Since $B$ prefers $Y$ to $X$, this is not a volatile pair and the matching is stable.

(b) Let $|E|$ be the number of different sex relationships between males and female. Let $e_m$ be the number of same-sex relationships among males, and $e_f$ the number of same-sex relationships among females. We are given that $e_m + e_f$ is 1% of all relationships,

$$
\frac{e_m + e_f}{|E|} = 0.01 \rightarrow e_m + e_f = 0.01 \times |E|.
$$

The total number of partners the men had is $|E| + 2e_m$ and similarly, the total number of partners the women had is $|E| + 2e_f$. So,

$$
average \ partners \ per \ male = \frac{|E| + 2e_m}{M},
$$

$$
average \ partners \ per \ female = \frac{|E| + 2e_f}{F}.
$$

Taking the ratio,

$$
\frac{average \ partners \ per \ male}{average \ partners \ per \ female} = \frac{|E| + 2e_m}{|E| + 2e_f} \times \frac{M}{F}.
$$

The two extremes are when $e_m = 0$ and when $e_f = 0$. When $e_m = 0$, $e_f = |E| \times \frac{0.01}{0.99}$ and

$$
average \ partners \ per \ male = \frac{|E| + 2e_f}{|E| + 2e_f} \times \frac{F}{M} \approx 1.012.
$$

When $e_f = 0$, $e_m = |E| \times \frac{0.01}{0.99}$ and

$$
average \ partners \ per \ male = \frac{|E| + 2e_m}{|E| + 2e_m} \times \frac{F}{M} \approx 1.0534.
$$

The average number of partners for men is $1.2\% - 5.3\%$ larger, depending on how the same sex relationships are distributed between males and females.

Pop Quiz 12.2. $(T_1, R_1), (T_2, R_2), (T_3, R_3), (T_4, R_4)$.

Exercise 12.3. We try proof by induction, as follows. In the induction step, take a left-vertex with minimum left degree and match it with a right-vertex, removing the two matched vertices. Let us now examine the residual graph. The degrees of some right-vertices decreases by one, but the maximum right-vertex degree could stay the same. What about left-vertices. Since we removed a right-vertex, if that right node was linked to other left-vertices, the degree of those left-vertices will decrease by 1. This means that the minimum left-vertex degree could decrease, and in so doing it could drop below the maximum right-vertex degree. Therefore, we may not be able to apply the induction hypothesis to the residual graph, and so the proof by induction falters. Hall’s theorem implies Corollary 12.2 on page 164, so Hall’s theorem is a stronger result. It is often easier to prove a stronger claim by induction. This is because we get to assume more in $P(n)$ which offsets having to prove more in $P(n+1)$.

Exercise 12.4. A matching is stable if there is no pair of matches that is volatile.
(a) If a woman \( w \) has more than one suitor, she chooses her favorite and the other suitors (at least one) cross \( w \) from their list. When there is at most one man under every woman’s balcony, we have a stalemate. For every non-stalemate round (at least one woman has more than one suitor), a man crosses a woman from a list. There are a total of \( n^2 \) women on all the lists (each woman appears once on each list). Therefore there cannot be more than \( n^2 \) non-stalemate rounds of dating because there will be no more women left to cross out.

Conclusion: After at most \( n^2 \) rounds of dating, each woman has at most one suitor.

(b) If a woman \( w \) ever gets wooed, then those suitors had \( w \) on the top of their current list. She picks her favorite who must come back the next round as \( w \) will remain on the top of that favorite’s list. By induction, she will always have a suitor.

(c) According to the dating ritual, \( m \) will continue to woo as long as there are uncrossed women on his list. Since \( m \) is not married at the end, it must be that \( m \) has been rejected by every woman, which means that he has wooed every woman, including \( w \).

Suppose that there is an unmarried man \( m \) at the end of the ritual. Then there is an unmarried woman \( w \) who was wooed at sometime by \( m \). By part (b), from that point on, \( w \) will always have a suitor and so must end up married, a contradiction. Therefore, every man is married at the end of the ritual (and therefore so too is every woman).

(d) Suppose \( w \) is at the \( i \)th position on \( m \)’s list.

(a) If \( m \) never wooed \( w \) then \( m \) could not have been rejected by all the top \( i - 1 \) candidates on \( m \)’s list. Therefore \( m \) was ultimately accepted by one of these top \( i - 1 \) candidates and ended up married to that better candidate: \( m \) prefers his current partner to \( w \).

(b) If \( m \) did woo \( w \), but is not married to \( w \), then \( w \) rejected \( m \) for someone better, \( m’ \).

From this point on, in the dating ritual, \( w \) will continue to accept only candidates who are at least as good as \( m’ \) because \( m’ \) will return to \( w \) unless someone better comes along and \( w \) rejects \( m’ \). Therefore, \( w \) will end up married to someone at least as good (in her view) as \( m’ \), who she prefers to \( m \): \( w \) prefers her current partner to \( m \).

That the marriages are stable is now immediate from (i) and (ii). Consider any pair of married couples \( (m, w) \) and \( (m’, w’) \). If \( m’ \) had wooed \( w \) then \( w \) prefers \( m \) to \( m’ \) and would not wish to switch to \( m’ \). If \( m’ \) had not wooed \( w \) then \( m’ \) prefers his current partner \( w’ \) to \( w \). Hence, the pair of married couples is not volatile.

**Pop Quiz 12.6.** The dating ends after two rounds.

**Dating Round 1:**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Z</td>
<td>Y</td>
<td>Z</td>
<td>B</td>
<td>A, X</td>
</tr>
<tr>
<td>2.</td>
<td>Y</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>X</td>
<td>Z</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Dating Round 2:**

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Z</td>
<td>Y</td>
<td>Z</td>
<td>C</td>
<td>B</td>
</tr>
<tr>
<td>2.</td>
<td>Y</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3.</td>
<td>X</td>
<td>Z</td>
<td>Y</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Pop Quiz 12.7.**

(a) \( R_1, R_2, R_3 \) form a “clique”, that is every pair has an edge between them. Therefore every vertex in this group must be colored a different color otherwise an edge will connect two vertices of the same color. Thus, we need at least \( 3 \) colors.

(b) Certainly you need at least one color, and \( n \) colors suffice by coloring each vertex a different color (no matter what the graph). So \( 1 \leq \chi(G) \leq n \). The graph with \( n \) isolated vertices needs can be colored with one color and \( K_n \), the complete graph on \( n \) vertices, requires \( n \) colors.

**Exercise 12.8.** The graph is an example of a leveled graph in which the nodes can be partitioned into levels \( \ell = 1, 2, 3, 4, \ldots \) and edges only exist between vertices in adjacent levels. In this case you can alternate colors between levels and get a valid 2-coloring. We show how to represent the graph as a leveled graph which immediately gives a 2-coloring. To order the vertices so that Greedy gives a 2-coloring, simply order the vertices by levels.
Exercise 12.9. Certainly if $V \leq 6$, then 6 colors are enough by coloring each vertex a different color. Therefore, we only need to consider $V > 6$ in which case $E \leq 3V - 6$.

(a) Suppose every node-degree is at least 6, then the sum of the node-degrees is at least $6V$, so

$$2E = \text{sum of node-degrees} \geq 6V.$$  

We conclude that $E \geq 3V > 3V - 6$, which contradicts $E \leq 3V - 6$. Therefore at least 1 vertex has degree of 5 or less.

(b) Start with a planar drawing of a graph and remove a vertex and its edges. The remaining edges do not cross in the drawing that remains (since initially they did not cross). Therefore the remaining graph is planar.

(c) We use induction on $V$, the number of vertices in the graph. If $V \leq 6$ then the claim is trivially true. Suppose the claim is true for any planar graph of $V$ vertices and consider any planar graph with $V + 1$ vertices. By (a), there is a node with degree at most 5. Remove this vertex and its edges. By (b) the remaining graph is planar and has $V$ vertices, so by the induction hypothesis this remaining graph is 6-colorable. Now add back the removed vertex, keeping the colors of the vertices obtained from the 6-coloring of the smaller graph. Among the 6 colors, there must be at least one free color for the node we added back because that vertex has at most 5 neighbors. Therefore, our graph with $V + 1$ vertices is 6-colorable. By induction, every planar graph is 6-colorable.

(In fact, every planar graph is 5-colorable. The same basic induction can be used for the proof, but you have to be a little more careful in the induction step. The 4-color theorem says that every planar graph is 4-colorable, and that is hard to prove.)

Chapter 13

Pop Quiz 13.1. $10 \times 9 \times 8 \times 7 \times \cdots \times 2 \times 1 = 10! = 3628800$.

Exercise 13.2. The outcome is one of two types: $HS_2$ where $S_2$ is the sum of two dice; or $TS_4$ where $S_4$ is the sum of 4 dice. $S_2 \in \{2, 3, \ldots, 12\}$ (11 choices) and $S_4 \in \{4, 3, \ldots, 24\}$ (21 choices). Using the sum rule, the experiment has $11 + 21 = 32$ possible outcomes.

Exercise 13.3. The committees are “named” (labeled): committee 1, committee 2, …, committee 16. That is, the committees are “distinguishible”.

(a) An assignment can be specified by $s_1 s_2 s_3 \cdots s_{100}$, where $s_i \in \{1, 2, \ldots, 16\}$ is the committee senator $i$ gets assigned. By the product rule, there are $16^{100}$ such sequences, which equals the number of ways each senator can be assigned to exactly one of 16 “named” committees. (You should ponder on what happens if the committees are not distinguishible.)

(b) A senator can be in 0 or 1 committee and the assignment can be specified by $s_1 s_2 s_3 \cdots s_{100}$, where $s_i \in \{0, 1, \ldots, 16\}$: $s_i = 0$ if senator $i$ is assigned to no committee; otherwise, $s_i$ senator $i$’s committee. By the product rule, there are $17^{100}$ such sequences, which equals the number of ways each senator can be assigned to at most one of 16 “named” committees.

STOP: do not bother with the solution of (c) if this is your first reading.

It is a hard problem, not essential for the rest of the chapter.

(c) The complication arises because by requiring that each committee is not empty we introduce a dependency between the senators, where as previously the senators can be assigned independently. For example, if the first 99 senators all get assigned to the first 15 committees, now the only available choice for $s_{100}$ is committee 16, otherwise that committee would be empty. Let us consider the case of 5 senators and 2 committees.

468
Exactly 1 committee per senator; no empty committee. When committees can be empty, there are \(2^n\) assignments. If all 5 senators are in either committee there is an empty committee, so these two assignments are not allowed. All others are allowed, so there are \(2^5 - 2\) ways.

The general case is when \(n\) senators are to be distributed into \(k\) “named” committees (in the literature you will often see this as \(n\) distinguishable objects being partitioned into \(k\) distinguishable non-empty sets). First consider the case where the committees are indistinguishable (that is not named). For example, with 5 senators and 3 committees, the following sequences are the same committee assignments if the committees are unlabeled:

12333 21333 31222 32111 23111 32111.

What matters is who is in a committee with whom. Let \(\{ ^n_k \}\) be the number of ways to create indistinguishable committees. Now we can label the committees: pick one of the \(k\) labels for the first committee, one of the remaining \(k - 1\) labels for the second committee and so on resulting in \(k \times (k - 1) \times \cdots \times 1\) ways to label the committees (product rule). So,

\[
\# \text{ ways to create } k \text{ non-empty "named" committees from } n \text{ senators} = k! \{ ^n_k \}.
\]

The numbers \(\{ ^n_k \}\) are known as Stirling numbers of the second kind,

\[
\{ ^n_k \} = \# \text{ ways to create partition } n \text{ labeled objects into } k \text{ non-empty unlabeled sets}.
\]

Stirling numbers are well studied. Here are some facts for you to verify (\(n \geq 1\), \(k \geq 1\)):

\[
\{ ^0_0 \} = 1 \quad \{ ^n_0 \} = 0 \quad \{ ^0_k \} = 0 \quad \{ ^n_{n-1} \} = \frac{1}{2}n(n-1) \quad \{ ^n_2 \} = 2^{n-1} - 1.
\]

The Stirling numbers \(\{ ^n_k \}\) satisfy a recurrence. To partition \(n\) objects into \(k\) sets: the first object can be in its own set and the other \(n - 1\) objects are partitioned into \(k - 1\) non-empty sets in \(\{ ^{n-1}_{k-1} \}\) ways; or, the first object is in a set with some other objects, in which case the other \(n - 1\) objects are partitioned into \(k\) non-empty subsets in \(\{ ^{n-1}_k \}\) ways which we multiply by the \(k\) to account for the \(k\) possible sets for the first object. Therefore,

\[
\{ ^n_k \} = k \{ ^{n-1}_{k-1} \} + \{ ^{n-1}_k \}.
\]

We leave the reader to use this recurrence and prove by induction that

\[
\{ ^n_k \} = \sum_{i=0}^{k} (-1)^{k-i} \frac{i^n}{i!(k-i)!}.
\]

To conclude,

\[
\# \text{ ways to create } k \text{ non-empty "named" committees from } n \text{ senators (one committee per senator)} = \sum_{i=0}^{k} (-1)^{k-i} \frac{k!}{i!(k-i)!} \cdot n.
\]

At most 1 committee per senator; no empty committee. We consider the problem by “brute-force”, first deciding the number of senators in no committees. There are 6 cases:

<table>
<thead>
<tr>
<th># senators not on a committee</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td># ways to pick the excluded senators</td>
<td>1</td>
<td>5</td>
<td>10</td>
<td>10</td>
<td>5</td>
<td>1</td>
</tr>
</tbody>
</table>

You can verify the number of ways to exclude \(k\) senators in forming the committees: there is 1 way to exclude 0 senators (all senators are involved in committees) and 1 way to exclude 5 senators (no senators are involved in committees); there are 5 ways to exclude 1 senator (5 possible senators to exclude) and similarly 5 ways to exclude 4 senators (5 possible senators to include). The number of ways to form a pair of senators is 10, so there are 10 ways to exclude 2 senators and similarly 10 ways to exclude 3 senators (select a pair to include). If you exclude 4 or 5 senators, then you cannot have both committees nonempty.

Conclusion: there are 4 types of committees: those that exclude 0,1,2 or 3 senators. Let’s count the number of ways to form 2 non-empty committees if you exclude 2 senators. So you use 3 senators. There are 10 ways to pick which 2 senators to exclude, and then there are \(2^3 - 2\) ways to form 2 non-empty committees using the remaining 3 senators. By the product
rule, there are \(10 \times (2^3 - 2)\) ways to form the two non-empty committees. Using this logic, we compute the entries in the following table for the number of committees that can be formed by excluding \(k\) senators, \(k = 0, 1, 2, 3\).

<table>
<thead>
<tr>
<th># committees</th>
<th># senators not on a committee</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(1 \times (2^3 - 2))</td>
</tr>
<tr>
<td>1</td>
<td>(5 \times (2^4 - 2))</td>
</tr>
<tr>
<td>2</td>
<td>(10 \times (2^3 - 2))</td>
</tr>
<tr>
<td>3</td>
<td>(10 \times (2^2 - 2))</td>
</tr>
</tbody>
</table>

Using the sum rule, the total number of committees we can form with at most 1 committee per senator and no empty committees is

\[
1 \times (2^5 - 2) + 5 \times (2^4 - 2) + 10 \times (2^3 - 2) + 10 \times (2^2 - 2) = 170.
\]

For the general case with \(n\) senators and \(k\) committees, we can leave out \(i\) senators and assign \(n - i\) of them to \(k\) non-empty committees in \(\frac{n!}{i!(n-i)!}\) ways, providing \(n - i \geq k\). You will see later that there are \(\frac{n!}{i!(n-i)!}\) ways in which to exclude \(i\) senators, so using the sum rule,

\[
\text{# ways to create } k \text{ non-empty "named" committees from } n \text{ senators (at most one committee per senator)} = \sum_{i=0}^{n-k} \frac{n!}{i!(n-i)!} \binom{n-i}{k}.
\]

**Pop Quiz 13.4.**

(a) To list all the sequences of length 6, prepend 0 and 1 to the length-5 sequences:

- 000000
- 000001
- 000010
- 000011
- 000100
- 000101
- 000110
- 000111
- 001000
- 001001
- 001010
- 001011
- 001100
- 001101
- 001110
- 001111
- 010000
- 010001
- 010010
- 010011
- 010100
- 010101
- 010110
- 010111
- 011000
- 011001
- 011010
- 011011
- 011100
- 011101
- 011110
- 011111
- 100000
- 100001
- 100010
- 100011
- 100100
- 100101
- 100110
- 100111
- 101000
- 101001
- 101010
- 101011
- 101100
- 101101
- 101110
- 101111
- 110000
- 110001
- 110010
- 110011
- 110100
- 110101
- 110110
- 110111
- 111000
- 111001
- 111010
- 111011
- 111100
- 111101
- 111110
- 111111

The color coding is: purple for 0 ones; green for 1 one; red for 2 ones; blue for 3 ones. You can count the sequences of each color to verify the first 6 entries in the row for \(n = 6\). The other entries follow by symmetry: if you take a sequence with 2 ones and flip every zero to one and vice-versa, you get a sequence with 4 ones. Thus, the number of sequences with 4 ones equals the number of sequences with 2 ones.

(b) We want \(\binom{10}{4}\), so we fill out our Pascal’s triangle table up to row \(n = 10\).

\[
\begin{array}{cccccccccc}
\binom{n}{k} & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \\
0 & 1 & & & & & & & & & & \\
1 & 1 & 1 & & & & & & & & & \\
2 & 1 & 2 & 1 & & & & & & & & \\
3 & 1 & 3 & 3 & 1 & & & & & & & \\
4 & 1 & 4 & 6 & 4 & 1 & & & & & & \\
5 & 1 & 5 & 10 & 10 & 5 & 1 & & & & & \\
6 & 1 & 6 & 15 & 20 & 15 & 6 & 1 & & & & \\
7 & 1 & 7 & 21 & 35 & 35 & 21 & 7 & 1 & & & \\
8 & 1 & 8 & 28 & 56 & 70 & 56 & 28 & 8 & 1 & & \\
9 & 1 & 9 & 36 & 84 & 126 & 126 & 84 & 36 & 9 & 1 & \\
10 & 1 & 10 & 45 & 210 & 252 & 210 & 120 & 45 & 10 & 1 & \\
\end{array}
\]

We highlighted the number we seek: \(\binom{10}{4} = 210\).

**Exercise 13.5.**

(a) (i) Using (13.2) on page 13.2,

\[
Q(n, k) = \sum_{j=0}^{n} Q(j, k-1) = Q(n, k-1) + \sum_{j=0}^{n-1} Q(j, k-1).
\]

The sum on the RHS is \(Q(n-1, k)\) by (13.2), so \(Q(n, k) = Q(n, k-1) + Q(n-1, k)\).

(ii) If there are no candies of color-1 in the goody bag, the goody bag is made up of \(n\) candies using \(k - 1\) colors: there are \(Q(n, k - 1)\) such goody bags. Or, there is at least 1 candy
of color-1. Place one candy of color-1 in the bag. The remaining \( n - 1 \) candies make up a “goody bag” using \( k \) colors, so there are \( Q(n-1,k) \) such goody bags. By the sum rule, the total number of goody bags is \( Q(n,k-1) + Q(n-1,k) \).

(b) The dashed diagonal produces the same numbers as row 5 in Pascal’s triangle. The next diagonal produces the numbers in Pascal’s triangle for row 6. Along the diagonal, \( n + k \) is constant. For the dashed diagonal, \( n + k = 6 \). The next diagonal has \( n + k = 7 \). So,

\[
\begin{align*}
n + k &= 6 \leftrightarrow \text{row 5 in Pascal’s triangle;} \\
n + k &= 7 \leftrightarrow \text{row 6 in Pascal’s triangle.}
\end{align*}
\]

The diagonal gives you the row in Pascal’s triangle, \( \text{row } = n + k - 1 \). And, \( k \) corresponds to the column in Pascal’s triangle \((k = 1 \text{ is column } 0)\), so \( k \) corresponds to column \( k - 1 \) in Pascal’s triangle. That is, \( m = n + k - 1 \) and \( \ell = k - 1 \) and our guess is

\[
Q(n,k) = \binom{n+k-1}{k-1}.
\]

Exercise 13.6.

(a) The first thing you do when using build up counting is to identify the object you are counting with a name and tinker. Let us denote by \( F(n) \) the number of subsets of \( \{1,2,\ldots,n\} \) that do not contain consecutive numbers. Now tinker with some small values of \( n \).

<table>
<thead>
<tr>
<th>( n )</th>
<th>subsets</th>
<th>( F(n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( \emptyset, {1} )</td>
<td>( F(1) = 2 )</td>
</tr>
<tr>
<td>2</td>
<td>( \emptyset, {1}, {2} )</td>
<td>( F(2) = 3 )</td>
</tr>
<tr>
<td>3</td>
<td>( \emptyset, {1}, {2}, {3}, {1,3} )</td>
<td>( F(3) = 5 )</td>
</tr>
</tbody>
</table>

Now, let us try to build such a subset \( S \). If \( S \) contains \( n \), then it cannot contain \( n - 1 \). So the remaining elements in \( S \) are a subset of \( 1,2,\ldots,n-2 \) not containing two consecutive numbers, and there are \( F(n-2) \) different such subsets (by definition of \( F(n) \)). If \( S \) does not contain \( n \), then the elements in \( S \) are a subset of \( 1,2,\ldots,n-1 \) and there are \( F(n-1) \) different such subsets. Those are the only options for \( S \), so by the sum rule,

\[
F(n) = F(n-1) + F(n-2).
\]

We can now compute \( F(20) \),

<table>
<thead>
<tr>
<th>( n )</th>
<th>( F(n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>13</td>
</tr>
<tr>
<td>5</td>
<td>21</td>
</tr>
<tr>
<td>6</td>
<td>34</td>
</tr>
<tr>
<td>7</td>
<td>55</td>
</tr>
<tr>
<td>8</td>
<td>89</td>
</tr>
<tr>
<td>9</td>
<td>144</td>
</tr>
<tr>
<td>10</td>
<td>233</td>
</tr>
<tr>
<td>11</td>
<td>377</td>
</tr>
<tr>
<td>12</td>
<td>610</td>
</tr>
<tr>
<td>13</td>
<td>987</td>
</tr>
<tr>
<td>14</td>
<td>1597</td>
</tr>
<tr>
<td>15</td>
<td>2584</td>
</tr>
<tr>
<td>16</td>
<td>4181</td>
</tr>
<tr>
<td>17</td>
<td>6765</td>
</tr>
<tr>
<td>18</td>
<td>10946</td>
</tr>
<tr>
<td>19</td>
<td>17711</td>
</tr>
<tr>
<td>20</td>
<td></td>
</tr>
</tbody>
</table>

There are 17,711 subsets of \( \{1,2,\ldots,20\} \) that do not contain consecutive numbers.

(b) Let us denote by \( G(n) \) the number of subsets of \( \{1,2,\ldots,n\} \) that contain at most 1 of any 3 consecutive numbers. Tinker with some small values of \( n \).

<table>
<thead>
<tr>
<th>( n )</th>
<th>subsets</th>
<th>( G(n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( \emptyset, {1} )</td>
<td>( G(1) = 2 )</td>
</tr>
<tr>
<td>2</td>
<td>( \emptyset, {1}, {2} )</td>
<td>( G(2) = 3 )</td>
</tr>
<tr>
<td>3</td>
<td>( \emptyset, {1}, {2}, {3} )</td>
<td>( G(3) = 4 )</td>
</tr>
<tr>
<td>4</td>
<td>( \emptyset, {1}, {2}, {3}, {4}, {1,4} )</td>
<td>( G(4) = 6 )</td>
</tr>
</tbody>
</table>

Now, let us try to build such a subset \( S \). If \( S \) contains \( n \), then it cannot contain \( n - 1 \) or \( n - 2 \). So the remaining elements in \( S \) are a subset of \( 1,2,\ldots,n-2 \) containing at most one of any three consecutive numbers, and there are \( G(n-3) \) different such subsets (by definition of \( G(n) \)). If \( S \) does not contain \( n \), then the elements in \( S \) are a subset of \( 1,2,\ldots,n-1 \) and there are \( G(n-1) \) different such subsets. Those are the only options for \( S \). By the sum rule,

\[
G(n) = G(n-1) + G(n-3).
\]

We can now compute \( G(20) \),

<table>
<thead>
<tr>
<th>( n )</th>
<th>( G(n) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>6</td>
<td>13</td>
</tr>
<tr>
<td>7</td>
<td>19</td>
</tr>
<tr>
<td>8</td>
<td>28</td>
</tr>
<tr>
<td>9</td>
<td>41</td>
</tr>
<tr>
<td>10</td>
<td>60</td>
</tr>
<tr>
<td>11</td>
<td>88</td>
</tr>
<tr>
<td>12</td>
<td>129</td>
</tr>
<tr>
<td>13</td>
<td>189</td>
</tr>
<tr>
<td>14</td>
<td>277</td>
</tr>
<tr>
<td>15</td>
<td>406</td>
</tr>
<tr>
<td>16</td>
<td>595</td>
</tr>
<tr>
<td>17</td>
<td>872</td>
</tr>
<tr>
<td>18</td>
<td>1278</td>
</tr>
<tr>
<td>19</td>
<td>1873</td>
</tr>
<tr>
<td>20</td>
<td>2745</td>
</tr>
</tbody>
</table>

There are 2,745 subsets of \( \{1,\ldots,20\} \) containing at most one of any three consecutive numbers.

(c) The method is the same, the problem is different. Let us denote by \( B(n) \) the number of
length-$n$ sequences not containing 001. Tinker with some small values of $n$.

<table>
<thead>
<tr>
<th>$n$</th>
<th>subsets</th>
<th>$B(n)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0, 1</td>
<td>$B(1) = 2$</td>
</tr>
<tr>
<td>2</td>
<td>00, 01, 10, 11</td>
<td>$B(2) = 4$</td>
</tr>
<tr>
<td>3</td>
<td>000, 010, 011, 100, 101, 110, 111</td>
<td>$B(3) = 7$</td>
</tr>
</tbody>
</table>

Let us now try to build a sequence $s$ of length $n$. It either starts with 1 or 0.

- $B(n-1)$ sequences
- $B(n-2)$ sequences
- 1 sequence

If $s$ starts with 1, what follows is any sequence of length $n-1$ that does not contain 001, and there are $B(n-1)$ of these. If $s$ starts with 0, there are two cases: the second bit is 1 in which case what follows is any sequence of length $n-2$ that does not contain 001, and there are $B(n-2)$ of these; the second bit is 0 in which case all remaining bits are 0, because otherwise the sequence contains 001. Therefore,

$$B(n) = B(n-1) + B(n-2) + 1.$$ 

We can now compute $B(20)$,

<table>
<thead>
<tr>
<th>$n$</th>
<th>$B(n)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
</tr>
<tr>
<td>5</td>
<td>20</td>
</tr>
<tr>
<td>6</td>
<td>33</td>
</tr>
<tr>
<td>7</td>
<td>54</td>
</tr>
<tr>
<td>8</td>
<td>88</td>
</tr>
<tr>
<td>9</td>
<td>159</td>
</tr>
<tr>
<td>10</td>
<td>2853</td>
</tr>
<tr>
<td>11</td>
<td>4180</td>
</tr>
<tr>
<td>12</td>
<td>7710</td>
</tr>
<tr>
<td>13</td>
<td>1596</td>
</tr>
<tr>
<td>14</td>
<td>232</td>
</tr>
<tr>
<td>15</td>
<td>376</td>
</tr>
<tr>
<td>16</td>
<td>609</td>
</tr>
<tr>
<td>17</td>
<td>986</td>
</tr>
<tr>
<td>18</td>
<td>17710</td>
</tr>
<tr>
<td>19</td>
<td>28656</td>
</tr>
<tr>
<td>20</td>
<td>4180</td>
</tr>
</tbody>
</table>

You should notice a similarity in our table for $B(n)$ and the table for $F(n)$ above. There are $28,656$ binary sequences of length $20$ that do not contain 001.

(d) Start small. For 2 players, there is just one way to configure the first round. Let $P(n)$ be the number of ways to configure the first round with 2n players. $P(1) = 1$. Now consider 2n players. The first player can pair with any of 2n-1 players leaving 2(n-1) players to be paired in $P(n-1)$ ways. Therefore, $P(n) = 2(n-1)P(n-1)$, and we have

<table>
<thead>
<tr>
<th>$n$</th>
<th>$P(n)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
</tr>
<tr>
<td>5</td>
<td>32</td>
</tr>
<tr>
<td>6</td>
<td>64</td>
</tr>
<tr>
<td>7</td>
<td>128</td>
</tr>
<tr>
<td>8</td>
<td>256</td>
</tr>
<tr>
<td>9</td>
<td>512</td>
</tr>
<tr>
<td>10</td>
<td>1024</td>
</tr>
<tr>
<td>11</td>
<td>2048</td>
</tr>
<tr>
<td>12</td>
<td>4096</td>
</tr>
<tr>
<td>13</td>
<td>8192</td>
</tr>
<tr>
<td>14</td>
<td>16384</td>
</tr>
<tr>
<td>15</td>
<td>32768</td>
</tr>
<tr>
<td>16</td>
<td>65536</td>
</tr>
<tr>
<td>17</td>
<td>131072</td>
</tr>
<tr>
<td>18</td>
<td>262144</td>
</tr>
<tr>
<td>19</td>
<td>524288</td>
</tr>
<tr>
<td>20</td>
<td>1048576</td>
</tr>
</tbody>
</table>

There are 2,027,025 different configurations for the first round matches.

**Exercise 13.7.**

(a) The claim is that $Q(n, k) = \binom{n+k-1}{k-1}$. We prove this by a “double induction”. We prove by induction on $k$, and within the induction on $k$, we use induction on $n$. Our claim is:

$$P(k) : Q(n,k) = \binom{n+k-1}{k-1}$$

for all $n \geq 0$.

We prove by induction that $P(k)$ is true for all $k \geq 1$.

The base case is $k = 1$ which claims $Q(n, 1) = \binom{n}{0} = 1$ which is true.

For the induction, assume $P(k)$. We show that

$$P(k+1) : Q(n, k+1) = \binom{n+k}{k}$$

for all $n \geq 0$.

When $n = 0$, $Q(0, k+1) = 1 \binom{k}{k}$. Let $n_*$ be the smallest $n$ for which $Q(n, k+1) \neq \binom{n+k}{k}$ (well ordering principle). Thus, $n_* > 0$. By the induction hypothesis, $Q(n_*, k) = \binom{n_*+k-1}{k-1}$.

Since $n_*$ is the smallest $n$ which fails, $Q(n_*-1, k+1) = \binom{n_*+k-1}{k}$.

By Exercise 13.5, $Q(n, k) = Q(n, k-1) + Q(n-1, k)$, therefore

$$Q(n_*, k+1) = Q(n_*, k) + Q(n_*-1, k+1)$$

$$= \binom{n_*+k-1}{k-1} + \binom{n_*+k}{k} = \binom{n_*+k}{k},$$

where in the last step we used the recursion $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$. The last expression shows that $n_*$ is not a counterexample, which is a contradiction. Therefore there is no smallest counterexample, and so $P(k+1)$ is true, concluding the proof by induction.

(b) The expression for $Q(n, k)$ follows by using (a) with $n + k - 1$ instead of $n$ and $k - 1$ instead of $k$. We prove that $\binom{n}{k} = n!/k!(n-k)!$. Define our claim,

$$P(n) : \binom{n}{k} = \frac{n!}{k!(n-k)!} \text{ for } 0 \leq k \leq n.$$
30. Solutions to Quizes & Exercises

We prove by induction that \( P(n) \) is true for all \( n \geq 1 \). First, we verify the base case \( n = 1 \), \( {n \choose 1} = 1 = 1!/0! \) and similarly \( \binom{1}{1} = 1 = 1!/1! \). For the induction, assume \( P(n) \) is true.

\[
\binom{n+1}{k} = \binom{n}{k} + \binom{n}{k-1} \quad \text{(recursion in (13.1))}
\]
\[
= \frac{n!}{k!(n-k)!} + \frac{n!}{(k-1)!(n-k+1)!} \quad \text{(induction hypothesis)}
\]
\[
= \frac{n!}{(k-1)!(n-k)!} \left( \frac{1}{k} + \frac{1}{n-k+1} \right)
\]
\[
= \frac{n!}{(k-1)!(n-k)!} \cdot \frac{n+1}{k(n-k+1)}
\]
\[
= \frac{k!(n+1-k)!}{k!(n-k)!} \quad \text{(n + 1)!}
\]

Therefore \( P(n+1) \) is true, and, by induction, \( P(n) \) is true for all \( n \geq 1 \).

(c) We denoted these numbers by \( F(n) \) in the solution to Exercise 13.7(a), where we showed that \( F(1) = 2, F(2) = 3 \) and \( F(n) = F(n-1) + F(n-2) \) (the Fibonacci recursion). We prove that \( F(n) = F_{n+2} \) by strong induction. The base cases \( n = 1, 2 \) are true because \( F_3 = 2 \) and \( F_4 = 3 \). For the induction step, we have that

\[
F(n+1) = F(n) + F(n-1) = F_{n+2} + F_{n+1} = F_{n+3}.
\]

(The first step is the recursion for \( F(n) \); the second is by the strong induction hypothesis; and, the third uses the Fibonacci recursion.) By induction, \( F(n) = F_{n+2} \) for \( n \geq 1 \).

(d) We denoted these numbers by \( B(n) \) in the solution to Exercise 13.7(c) (we showed that \( B(1) = 2, B(2) = 4 \)). For these two base cases, \( B(n) = F_{n+3} - 1 \). For the strong induction step,

\[
B(n+1) = B(n) + B(n-1) + 1 = F_{n+3} + F_{n+2} - 1 + 1 = F_{n+4} - 1.
\]

(The first step is the recursion for \( B(n) \); the second is by the strong induction hypothesis; and, the last step uses the Fibonacci recursion.) By induction, \( B(n) = F_{n+3} - 1 \) for \( n \geq 1 \).

Pop Quiz 13.8.

(a) Everything in \( A \) maps to one element of \( B \), \( f(1) = f(2) = f(3) = f(4) = 2 \).
(b) Trick question. 1-to-1 but not onto can only be done if \( |A| < |B| \).
(c) Trick question. Not 1-to-1 but onto can only be done if \( |A| > |B| \).
(d) \( F(1) = 2; F(2) = 3; F(3) = 4; F(4) = 5 \).

Pop Quiz 13.9.

(a) \{3, 4, 5, 6\}
(b) 000101000000
(c) Let \( 0^n \cdot \cdot \cdot 0^i \cdot 1 \cdot 0^k \) be a binary sequence of length \( n+2 \) with 2 ones. So, \( i, j, k \geq 0 \) and \( i + j + k = n \). The sequence corresponds to the bag \( \{i, j, k\} \) containing \( i \) red candies, \( j \) blue candies and \( k \) green candies.

Clearly this correspondence is 1-to-1 because two different sequences will have different triples \( (i, j, k) \) which maps to different candy bags. For any candy bag, we can construct the sequence, so the mapping is onto, hence a bijection.

Exercise 13.10.

(a) \( x_i \) is the number of candies of color \( i \) and since there are 10 candies, \( \sum x_i = 10 \).

Any goodiy bag with the 10 candies gives non-negative \( x_i \)'s which sum to 10. Any non-negative integer solution to \( x_1 + \cdots + x_4 = 10 \) gives a candy bag with \( x_i \) of candy \( i \). We have a bijection between the candy bags and the non-negative solutions, that is

\[
Q(10, 4) = \binom{10+4-1}{4-1} = \binom{10}{3} = \text{number of non-negative solutions to } x_1 + \cdots + x_4 = 10.
\]

(b) Let \( y_1 = x_1 - 1 \). Then \( y_1 \) are non-negative and \( y_1 + \cdots + y_4 = x_1 + \cdots + x_4 - 4 = 6 \). Every non-negative solution to \( y_1 + \cdots + y_4 = 6 \) gives a positive solution to \( x_1 + \cdots + x_4 = 10 \) and vice versa. Therefore, the answer is \( Q(6, 4) = \binom{6+4-1}{4-1} = \binom{9}{3} \).

(c) Introduce a dummy variable \( x_5 = 10 - (x_1 + \cdots + x_4) \), \( x_5 \geq 0 \) and \( x_1 + \cdots + x_5 = 10 \). Every non-negative solution to \( x_1 + \cdots + x_5 = 10 \) gives a non-negative solution to \( x_1 + \cdots + x_4 \leq 10 \), so the answer is \( Q(10, 5) = \binom{10+5-1}{5-1} = \binom{14}{4} \).
30. Solutions to Quizes & Exercises

(d) For the binary sequence $b_1 \cdots b_{10}$ construct the subset $A$ containing the elements where $b_i = 1$,
\[ A = \{ x_i \mid b_i = 1 \}. \]

Every binary sequence with 3 ones gives corresponds to a unique subset $A$ with 3 elements and for every 3-subset $A$ we can construct the binary sequence with 3 ones, so we have a bijection. Therefore the number of such subsets equals the number of binary sequences with 3 ones, which is $\binom{10}{3}$. In general, the number of $k$-subsets of an $n$-element set is $\binom{n}{k}$.

(e) A 3-subset corresponds uniquely to its complement (a 7-subset) and vice versa. Since we have a bijection from 3-subsets to 7-subsets, $\binom{10}{3} = \binom{10}{7}$. In general $\binom{n}{k} = \binom{n}{n-k}$.

(f) The bijection is exactly the same as in (d), but to $n$-bit binary sequences with $k$ ones. The ones indicate which elements are in the subset.

(g) Use the bijection in (e) from a $k$-subset to its complement, an $(n-k)$-subset.

(h) We proved using the product rule on page 184 that there are $2^n$ binary sequences of length $n$. Each sequence uniquely describes a subset (the ones correspond to the elements in the subset), so there are $2^n$ subsets of a set (see also Example 13.1 about Senate committees on page 186). Here is another way to count the subsets using the sum rule:
\[ |\{\text{subsets}\}| = |\{\text{subsets of size 0}\}| + |\{\text{subsets of size 1}\}| + \cdots + |\{\text{subsets of size n}\}|. \]

From parts (d) and (f), the number of subsets of size $k$ is $\binom{n}{k}$, therefore
\[ 2^n = \sum_{k=0}^{n} \binom{n}{k}. \]

This problem illustrates one of the fundamental techniques for establishing that two combinatorial expressions are equal: count a set in two different ways; the answers must be equal.

In our example one way of counting gave $2^n$, and the other gave $\sum_{k=0}^{n} \binom{n}{k}$.

Pop Quiz 13.11. This problem is actually more complicated than it appears at first sight. The king can be placed in one of 64 positions, removing 15 possible row-column squares for the queen. How many diagonal squares are removed? It depends on where the king is.

On the left we show two possible king positions. The king on $b_2$ covers 9 diagonal squares (excluding its own square), but the king on $f3$ covers 11 diagonal squares. You should verify for the figure on the right that the positions on the black box all cover 9 diagonal squares and those on the red box all cover 11 diagonal squares. Our observation suggests that we should use the sum rule with four types of positions for the king: the outer-most ring of size 28 (covering 7 diagonal squares) to inner-most ring of size 4 (covering 13 diagonal squares)

<table>
<thead>
<tr>
<th>ring size</th>
<th>28</th>
<th>20</th>
<th>12</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>diagonal squares covered</td>
<td>7</td>
<td>9</td>
<td>11</td>
<td>13</td>
</tr>
</tbody>
</table>

The number of positions available to the queen, given the position of the king is
\[ 64 - 15 \text{ row-column squares} - \text{number of diagonal squares covered}. \]

Denoting the rings by 0,1,2,3 (ring-0 is outermost). Here are our observations:

<table>
<thead>
<tr>
<th>type of king position</th>
<th>ring-0</th>
<th>ring-1</th>
<th>ring-2</th>
<th>ring-3</th>
</tr>
</thead>
<tbody>
<tr>
<td>number of possible king positions</td>
<td>28</td>
<td>20</td>
<td>12</td>
<td>4</td>
</tr>
<tr>
<td>diagonal squares covered</td>
<td>8</td>
<td>10</td>
<td>12</td>
<td>14</td>
</tr>
<tr>
<td>number of possible queen positions</td>
<td>42</td>
<td>40</td>
<td>38</td>
<td>36</td>
</tr>
</tbody>
</table>

Using the product rule within the types of king positions and the sum rule to add up the positions of each type to get the total number of possible positions,
\[ \text{number of possible positions} = 28 \times 42 + 20 \times 40 + 12 \times 38 + 4 \times 36 = 2576. \]
For practice, consider a general \( n \times n \) board. First, if the queen cannot be on the same row and column as the king, the number of ways to specify the sequence \( c_{KTQ} \) is 
\[
n \times n \times (n - 1) \times (n - 2) = n^2(n - 1)^2.
\]
If the queen cannot be on the same diagonal, the number of row-columns squares covered by the king is \( 2n - 1 \). In the outermost ring with \( 4n - 4 \) squares, the number of diagonals covered is \( n - 1 \), so the number of possible queen positions is \( n^2 - (2n - 1) - (n - 1) \). Each time you move in by one ring, the number of possible king positions decreases by 8 and the number of diagonals covered increases by 2, so the number of possible queen positions decreases by 2. Thus, in ring-i, the number of possible king positions is \( 4n - 4 - 8i \) and queen positions is \( n^2 - 3n + 2 - 2i = (n - 1)(n - 2) - 2i \). The number of rings is \( n/2 \) when \( n \). When \( n \) is odd, the number of rings is \( (n - 1)/2 \) plus the single center square. Using the product and sum rule for the rings,
\[
\text{number of possible positions} = \sum_{i=0}^{k} (4n - 4 - 8i)((n - 1)(n - 2) - 2i).
\]
where \( k = n/2 - 1 \) if \( n \) is even and \( k = (n - 1)/2 - 1 \) when \( n \) is odd (when \( n \) is odd, there is also the center square to consider). We use techniques from Chapter 9 to compute the sum.
\[
\text{number of possible positions} = 4 \sum_{i=0}^{k} (n - 1 - 2i)((n - 1)(n - 2) - 2i)
\]
\[
= 4 \left[ \sum_{i=0}^{k} (n - 1)^2(n - 2) - 2 \sum_{i=0}^{k} (n - 1)^2i + 4 \sum_{i=0}^{k} i^2 \right]
\]
\[
= 4 \left[ (k + 1)(n - 1)^2(n - 2) - k(k + 1)(n - 1)^2 + 4 \frac{k(k + 1)(2k + 1)}{6} \right]
\]
When \( n \) is even, we plug in \( k = n/2 - 1 \) to get 
\[
\text{number of possible positions} = \frac{1}{4}n(n - 1)(n - 2)(3n - 1).
\]
When \( n \) is odd, we plug in \( k = (n - 1)/2 - 1 \) and add the \( n^2 - 4n + 3 \) positions with the king on the center square. The resulting formula is the same.

**Pop Quiz 13.12.** To specify the positions of the 8 castles, we specify the sequence 
\[
(c_1r_1)(c_2r_2)(c_3r_3)(c_4r_4)(c_5r_5)(c_6r_6)(c_7r_7)(c_8r_8)
\]
for the column and row of each castle. For the columns, there are \( 8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 8! \) ways. For the rows, there are \( 8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 8! \) ways. For the rows and columns, by the product rule, there are \( (8!)^2 \) ways. Consider one such sequence, for example, \( (a1)(b2)(c3)(d4)(e5)(f6)(g7)(h8) \). If we reorder some positions, for example to \( (b2)(a1)(c3)(d4)(e5)(f6)(g7)(h8) \), we get a different sequence but the same position. There are \( 8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 8! \) possible reorderings of this position sequence, so this means that every position corresponds to \( 8! \) different sequences, a 1-to-8! mapping. By the multiplicity rule, there are \( 8! \) times as many sequences as there are positions. So the number of positions is \( (8!)^2/8! = 8! \times 40320 \).

Another way to quickly get this result is to observe that we may specify the position with the rows increasing. So every position is specified by a sequence of the form 
\[
\]
That is, we only get to choose the columns, and there are \( 8! \) ways to do that.

**Exercise 13.13.**
(a) A poker hand is a subset of 5 cards from 52. The number of poker hands equals the number of possible subsets which is \( \binom{52}{5} = 52!/(5! \times 47!) = 2598960 \).
(b) The idea for such problems is give a sequence of instructions that uniquely constructs the object. Your instructions must unambiguously construct the object. Effectively, you construct a bijection between sequences of instructions and the objects. Now count the sequences. Here is a “recipe” to construct a 4-of-a-kind poker hand:
1: Choose a value \( v \) and pick all four cards of value \( v \): Ace, Diamond, Heart, and Spade.
2: Choose one of the other cards of a different value \( c \).
The sequence \( vc \) completely specifies the 4-of-a-kind. Change any part of the sequence and you get a different 4-of-a-kind hand. We have a bijection between sequences \( vc \) and 4-of-a-kind hands. Counting the sequences is “easy”. There are 13 possible choices for \( v \) and for each
choice of \( v \) there are 48 choices for \( c \). By the product rule,
\[
|\{4-of-a-kind hands\}| = |\{sequences \( vc \)\}| = 13 \times 48 = 624.
\]

(c) To construct a flush, here is a recipe:
1. Pick the suit \( s \), either \( \heartsuit \), \( \diamondsuit \), or \( \spadesuit \).
2. Choose a set of 5 values \( V = \{v_1, v_2, v_3, v_4, v_5\} \) from the 13 values in suit \( s \).

The sequence \( sV \) completely specifies the flush. There are 4 choices for \( s \). Given \( s \), \( V \) is a 5-subset of the 13 values, so there are \( \binom{13}{5} \) choices for this subset. By the product rule,
\[
|\{flushes\}| = |\{sequences sV\}| = 4 \times \binom{13}{5} = 5148.
\]

(d) To construct a full-house, here is a recipe:
1. Choose a value \( v_1 \).
2. Choose \( T_1 = \{s_1, s_2, s_3\} \), a set of 3 suits having value \( v_1 \).
3. Choose a second value \( v_2 \neq v_1 \).
4. Choose \( T_2 = \{s_1, s_2\} \), a set of 2 suits having value \( v_2 \).

The sequence \( v_1 T_1 v_2 T_2 \) completely specifies the full-house. There are 13 choices for \( v_1 \); given \( v_1, T_1 \), \( T_1 \) is a 3-subset of the 4 suits, which can be picked in \( \binom{4}{3} \) ways; given \( v_1, T_1, v_2 \) has 12 choices (since \( v_2 \neq v_1 \)); given \( v_1, T_1, v_2, T_2 \) is a 2-subset of the 4 suits, which can be picked in \( \binom{4}{2} \) ways. By the product rule,
\[
|\{full-houses\}| = |\{sequences v_1 T_1 v_2 T_2\}| = 13 \times \binom{4}{3} \times 12 \times \binom{4}{2} = 3744.
\]

(e) To construct a 3-of-a-kind, here is a recipe:
1. Choose a value \( v \).
2. Choose \( T = \{s_1, s_2, s_3\} \), a set of 3 suits having value \( v \).
3. Choose \( c_1 \), a card of value \( v \neq v \).
4. Choose \( c_2 \), a card of value \( v_2 \neq v \) or \( v_1 \).

Let’s count the sequences \( vT c_1 c_2 \): there are 13 choices for \( v \); given \( v \), \( T \) is a 3-subset of the 4 suits, which can be picked in \( \binom{4}{3} \) ways; given \( v_1 \neq v \); there are 44 choices for \( c_2 \) (because \( v_2 \neq v \) or \( v_1 \)). By the product rule,
\[
|\{3-of-a-kinds\}| = |\{sequences vT\}| = 13 \times \binom{4}{3} \times 44 = 109842.
\]

WRONG! The mistake is similar to the issue with counting positions of two castles (indistinguishable pieces) on a chessboard versus a king and a queen (distinguishable pieces). The problem is with the last two cards in the 3-of-a-kind. (\( \spadesuit A, \heartsuit A, \spadesuit A, \heartsuit 7, \spadesuit 2 \)) and (\( \spadesuit A, \heartsuit A, \spadesuit A, \heartsuit 2, \heartsuit 7 \)) are the same hand. However the two sequences \( vT c_1 c_2 \) are different. So two different sequences map to the same hand: we do not have a 1-to-1 mapping.

There are many ways to resolve this problem. Observe that every hand maps to two sequences. Using the multiplicity rule, there are twice as many sequences as hands: the number of hands is 109842/2 = 54912. Alternatively, we view the remaining 48 cards in some order and pick \( c_1 c_2 \) with \( c_1 < c_2 \) (as we did with the castle positions). The number of ways to pick \( c_1 < c_2 \) is half the number of ways to pick \( c_1 c_2 \) (the other half have \( c_1 > c_2 \)). The systematic route is to give a recipe that uniquely constructs a 3-of-a-kind hand:
1. Choose a value \( v \).
2. Choose \( T = \{s_1, s_2, s_3\} \), a set of 3 suits having value \( v \).
3. Choose a pair of values \( V = \{v_1, v_2\} \) from the remaining 12 values.
4. Choose \( x_1 \), a suit from value \( v_1 \).
5. Choose \( x_2 \), a suit from value \( v_2 \).

Now, the sequence \( vT V x_1 x_2 \) uniquely constructs a 3-of-a-kind hand and we can count the sequences. There are 13 choices for \( v \); given \( v \), \( T \) is a 3-subset of the 4 suits, which can be picked in \( \binom{4}{3} \) ways; \( V \) is a 2-subset of the remaining 12 values, with \( \binom{12}{2} \) choices; given \( V, x_1 \) and \( x_2 \) each have 4 choices. By the product rule,
\[
|\{3-of-a-kinds\}| = |\{sequences vT V x_1 x_2\}| = 13 \times \binom{4}{3} \times \binom{12}{2} \times 4 \times 4 = 54912.
\]

(f) To construct a two-pair, here is a recipe:
1. Choose \( V = \{v_1, v_2\} \) the values for each pair.
2. Choose \( S_1 = \{s_1, s_2\} \), a set of 2 suits having value \( v_1 \).
3. Choose \( S_2 = \{s_1, s_2\} \), a set of 2 suits having value \( v_2 \).
4. Choose a the 5th card \( c \) from the 44 not of value \( v_1, v_2 \).

476
To count the sequences $\forall S_1, S_2$: there are $\binom{13}{2}$ choices for the 2-subset of values from 13; $S_1$ is a 2-subset of the 4 suits, which can be picked in $\binom{4}{2}$ ways; similarly, there are $\binom{4}{2}$ ways to pick $S_2$; lastly, there are 44 choices for $c$. By the product rule,
\[
|\{\text{two-pairs}\}| = |\{\text{sequences } \forall S_1, S_2, c\}| = \binom{13}{2} \times \binom{4}{2} \times 44 = 123552.
\]

**Chapter 14**

**Exercise 14.1.**
(a) We have a sequence of length $n = 8$ with $3 \emptyset, 2 \emptyset, 1 \emptyset$ and $1 \emptyset$. The number of sequences is $\binom{8}{3,2,1,1,1} = \frac{8!}{3!2!1!1!1!} = 3360$.

(b) A bouquet is exactly analogous to a goody-bag. The bouquet has 36 objects of 4 colors. The number of monomials in $x_1 x_2 x_3 x_4$ is the number of different monomials, which can be picked in $\binom{36}{4}$ ways.

(c) Let the students be $s_1 s_2 \cdots s_{25}$.

(i) There are 5 types of students: those assigned to cooking, cleaning, laundry, entertainment, groceries. There are 5 of each type of student, so we need a sequence of length 25 with 5 of each type. This can be done in $\binom{25}{5,5,5,5,5}$ ways.

\[
\left(\binom{25}{5,5,5,5,5}\right) = \frac{25!}{5!5!5!5!5!5!} = \frac{25!}{(5!)^5} = \frac{25!}{125000} = 25! 
\]

(We used Stirling’s formula, $n! \approx \sqrt{2\pi n} n^n e^{-n}$. The exact answer is about $6.234 \times 10^{14}$.)

(ii) For each task we pick a subset of 5 students to perform the task, which can be done in $\binom{25}{5}$ ways. By the product rule, the number of ways to perform the tasks is $(\binom{25}{5})^5 = 35130^5 \approx 4.2 \times 10^{25}$.

**Exercise 14.2.**
(a) For this problem we use the binomial and multinomial theorems:

(i) We want the coefficient of $1^4 x^3$ which is $\binom{3}{1} = 126$.

(ii) The coefficient of $(2x)^3 (3y)^1$ is $\binom{7}{2} = 35$ which gives $35(2x)^3 (3y)^1 = 35 \cdot 2^3 \cdot 3x^3 y$.

(iii) $x^4 y^8$ is the coefficient of $(x^1 (x^2 ) (y^2 ))^4$ which is $\binom{7}{1, 2, 5} = 105$.

(b) Monomials in $(x + y)^n$ are $x^i y^j$. There are $n + 1$ possible $i$, where $i = 0, 1, \ldots, n$, and given $i$, $j = n - i$. So, the number of different monomials is $n + 1$.

For $(x + y + z)^n$, the monomials are specified by $(i, j, k)$ where $i$ is the power of $x$, $j$ is the power of $y$ and $k$ is the power of $z$: $i$ ranges from 0 to $n$; given $i$, $j$ ranges from 0 to $n - i$ which is $(n - i + 1)$ choices for $j$; given $i$ and $j$, $k$ is $n - i - j$. So the number of monomials is $\sum_{i=0}^{n} (n - i + 1) = (n + 1) + n + \cdots + 1 = \frac{1}{2} (n + 1)(n + 2)$.

(c) The terms are $n$-sequences with tokens $a_1, \ldots, a_k$, so there are $k^n$ terms. The number of sequences with $i_1 a_1 s, i_2 a_2 s, \ldots, i_k a_k s$ is $\binom{n}{i_1, i_2, \ldots, i_k}$ and each equals $a_1^{i_1} a_2^{i_2} \cdots a_k^{i_k}$, where $i_1 \geq 0, \ldots, i_k \geq 0$ and $i_1 + \cdots + i_k = n$. We get the multinomial theorem:

\[
(a_1 + a_2 + \cdots + a_k)^n = \sum_{i_1, i_2, \ldots, i_k = 0}^{n} \binom{n}{i_1, i_2, \ldots, i_k} a_1^{i_1} a_2^{i_2} \cdots a_k^{i_k}.
\]

(d) The result is immediate from setting $a_1 = a_2 = \cdots = a_k = 1$ in part (c).

**Pop Quiz 14.3.** For convenience we repeat the derivation here.

\[
|A_1 \cup A_2 \cup A_3| \overset{(\text{a})}{=} |A_1 \cup A_2| + |A_3| - |(A_1 \cup A_2) \cap A_3|
\]
\[
\overset{(\text{b})}{=} |A_1| + |A_2| + |A_3| - |A_1 \cap A_2| - |(A_1 \cap A_3) \cup (A_2 \cap A_3)|
\]
\[
\overset{(\text{c})}{=} |A_1| + |A_2| + |A_3| - |A_1 \cap A_2| - (|A_1 \cap A_3| + |A_2 \cap A_3| - |A_1 \cap A_3 \cap A_2 \cap A_3|)
\]
\[
\overset{(\text{d})}{=} |A_1| + |A_2| + |A_3| - |A_1 \cap A_2| - |A_1 \cap A_3| - |A_2 \cap A_3| + |A_1 \cap A_2 \cap A_3|.
\]

477
Exercise 14.4.

(a) Let $B = A_1 \cup A_2$. Using inclusion-exclusion for two sets, $|B \cup A_3| = |B| + |A_3| - |B \cap A_3|$.

(b) Apply two set inclusion-exclusion to $|A_1 \cup A_2|$ and use the distributive property of intersection, 
$$(A_1 \cup A_2) \cap A_3 = (A_1 \cap A_3) \cup (A_2 \cap A_3)$$
(see Figure 2.1 on page 3).

(c) Let $A = A_3 \cap A_4$ and $B = A_2 \cap A_3$ and apply two set inclusion-exclusion to $|A \cup B|$.

(d) $|A_1 \cap A_3 \cap A_2 \cap A_4| = |A_1 \cap A_2 \cap A_3|$.

(i) We give a proof that will generalize to the induction step for the general case.

$$|A_1 \cup A_2 \cup A_3 \cup A_4| = |A_1 \cup A_2 \cup A_3| + |A_4| + |A_1 \cap A_2 \cap A_3 \cap A_4|$$

$$= |A_1 \cup A_2 \cup A_3| + |A_4| - |(A_1 \cup A_2 \cup A_3) \cap A_4|$$

We can apply 3-set inclusion-exclusion to the first and third terms:

$$|A_1 \cup A_2 \cup A_3| = \sum_{k=1}^{3} (-1)^{k+1} \cdot (\text{sum of } k\text{-way intersections involving } A_1, A_2, A_3)$$

$$|(A_1 \cap A_4) \cup (A_2 \cap A_4) \cup (A_3 \cap A_4)|$$

$$= \sum_{k=1}^{3} (-1)^{k+1} \cdot (\text{sum of } k\text{-way intersections involving } A_1 \cap A_4, A_2 \cap A_4, A_3 \cap A_4)$$

Crucial observation: a $k$-way intersection involving $A_1 \cap A_4, A_2 \cap A_4, A_3 \cap A_4$ is the intersection of $A_4$ with the corresponding $k$-way intersection involving $A_1, A_2, A_3$. For example, $(A_1 \cap A_4) \cap (A_2 \cap A_4) = A_1 \cap A_2 \cap A_4$. We conclude that

$$|A_1 \cup A_2 \cup A_3 \cup A_4|$$

$$= \sum_{k=1}^{3} (-1)^{k+1} \cdot (\text{sum of } k\text{-way intersections involving } A_1, A_2, A_3)$$

$$+ |A_4| + \sum_{k=1}^{3} (-1)^{k+2} \cdot (\text{sum of } A_4 \cap k\text{-way intersections involving } A_1, A_2, A_3)$$

The summands in the last term are the $k + 1$-way intersections involving $A_4$. The summands in the first term are the $k$-way intersections that do not include $A_4$.

$$|A_1 \cup A_2 \cup A_3 \cup A_4|$$

$$= \sum_{k=1}^{3} (-1)^{k+1} \cdot (\text{sum of } k\text{-way intersections not involving } A_4)$$

$$+ |A_4| + \sum_{k=1}^{3} (-1)^{k+2} \cdot (\text{sum of } (k + 1)\text{-way intersections involving } A_4)$$

The last two terms are all $k$-way intersections that involve $A_4$ with $k = 1, \ldots, 4$. So,

$$|A_1 \cup A_2 \cup A_3 \cup A_4|$$

$$= \sum_{k=1}^{3} (-1)^{k+1} \cdot (\text{sum of } k\text{-way intersections not involving } A_4)$$

$$+ \sum_{k=1}^{4} (-1)^{k+1} \cdot (\text{sum of } k\text{-way intersections involving } A_4)$$

Summing over $k$-way intersections involving $A_4$ and those not involving $A_4$ amounts to summing over all $k$-way intersections:

$$|A_1 \cup A_2 \cup A_3 \cup A_4| = \sum_{k=1}^{4} (-1)^{k+1} \cdot \text{(sum of } k\text{-way intersections)}$$
(ii) The proof for general \( n \) by induction mimics the proof for \( n = 4 \). The base case is \( n = 2 \), which we have already established. For the induction step, consider any \( n + 1 \) sets.

\[
|A_1 \cup A_2 \cup A_3 \cup \cdots \cup A_n \cup A_{n+1}| = |A_1 \cup A_2 \cup \cdots \cup A_n| + |A_{n+1}| - |(A_1 \cup A_2 \cup \cdots \cup A_n) \cap A_{n+1}|
\]

The first term, by the induction hypothesis, sums over the \( k \)-way intersections not involving \( A_{n+1} \). The last two terms sums over the \( k \)-way intersections involving \( A_{n+1} \).

\[
= \sum_{k=1}^{n} (-1)^{k+1} \cdot \text{(sum of } k \text{-way intersections not involving } A_{n+1})
\]

\[
+ \sum_{k=1}^{n+1} (-1)^{k+1} \cdot \text{(sum of } k \text{-way intersections involving } A_{n+1})
\]

That is, the formula holds for a union of \( n + 1 \) sets, and by induction for all \( n \geq 2 \).

(b) Let \( A_2, A_3, A_5, A_7 \) be the sets of numbers from 1 to 2015 that are divisible by 2, 3, 5, 7 respectively. We want \( 2015 - |A_2 \cup A_3 \cup A_5 \cup A_7| \). Here are two facts:

**Lemma 30.5.** There are \([ n/k ]\) numbers from 1 to \( n \) that are divisible by \( k \).

**Lemma 30.6.** \( x \) is divisible by \( d_1, d_2, \ldots, d_k \) if and only if \( x \) is divisible by the least common multiple of \( d_1, d_2, \ldots, d_k \).

To compute \( |A_2 \cup A_3 \cup A_5 \cup A_7| \), we need the sizes of all the \( k \)-way intersections. For example, \( |A_2| = \lceil 2015/2 \rceil \) and \( |A_2 \cap A_3| = \lceil 2015/6 \rceil \) because \( A_2 \cap A_3 \) contains numbers divisible by 2 and 3, and hence by 6. Define \( A_1 = A_2 \cap A_3 \) and similarly \( A_{ijk} \) and \( A_{ijkl} \). We have:

\[
\begin{align*}
|A_1| &= 1007; \\
|A_2| &= 671; \\
|A_3| &= 403; \\
|A_5| &= 287. \\
|A_{23}| &= 335; \\
|A_{25}| &= 201; \\
|A_{35}| &= 143; \\
|A_{235}| &= 67; \\
|A_{237}| &= 49; \\
|A_{257}| &= 28; \\
|A_{357}| &= 9.
\end{align*}
\]

Therefore, for \( |A_2 \cup A_3 \cup A_5 \cup A_7| \) we get:

\[
1007 + 671 + 403 + 287 - 335 - 201 - 143 - 134 - 95 - 67 + 49 + 28 + 9 = 1557.
\]

We conclude that \( 2015 - 1557 = 458 \) numbers are not divisible by any of \( \{2, 3, 5, 7\} \).

(c) We count the ways to distribute the hats so that some man gets the right hat and subtract from \( 4! \), the number of ways to distribute the hats. Let \( A_1 \) be the orderings in which man 1 gets his correct hat; similarly define \( A_2, A_3, A_4 \). The number of ways in which some man gets the right hat is \( |A_1 \cup A_2 \cup A_3 \cup A_4| \). For inclusion-exclusion, we need the intersections. Let \( A_{12} = A_1 \cap A_2 \) be the number of ways men 1 and 2 get the correct hats; similarly define \( A_{13}, A_{14}, A_{23}, A_{24}, A_{34} \) and so on. Now, \( |A_i| = 3! \) (3! ways to distribute the other 3 hats after giving man \( i \) his hat). Similarly, \( |A_{ij}| = 2! \) because there are 2! ways to distribute the other 2 hats after giving men \( i \) and \( j \) their hats; \( |A_{ijkl}| = 1! \); and, \( |A_{ijkl}| = 0! \). We have:

\[
\begin{align*}
|A_1| &= |A_2| = |A_3| = |A_4| = 6. \\
|A_{12}| &= |A_{13}| = |A_{14}| = |A_{23}| = |A_{24}| = |A_{34}| = 2. \\
|A_{123}| &= |A_{124}| = |A_{134}| = |A_{234}| = 1. \\
|A_{1234}| &= 1.
\end{align*}
\]

Applying the inclusion-exclusion formula,

\[
|A_1 \cup A_2 \cup A_3 \cup A_4| = 4 \times 6 - 6 \times 2 + 4 \times 1 - 1 = 15.
\]

The answer is \( 4! - 15 = 9 \). Distributing \( n \) objects so that no object goes into its correct spot is a derangement. Example 14.4 on page 206 discusses how to count derangements of \( n \) objects.
(a) Let $A_{12}$ be the permutations containing 12 and $A_{24}$ the permutations containing 24. There are 10! passwords. The invalid passwords are in $A_{12} \cup A_{24}$, so the number of valid passwords is $10! - |A_{12} \cup A_{24}|$. By inclusion-exclusion,

$$|A_{12} \cup A_{24}| = |A_{12}| + |A_{24}| - |A_{12} \cap A_{24}|.$$  

To count passwords containing 12, treat 12 as a single token. We want a permutation of 0123456789. There are 9! permutations, i.e. 9! such passwords. Similarly, there are 9! passwords containing 24. Therefore $|A_{12}| = |A_{24}| = 9!$. The passwords containing 12 and 24 must contain 124. Treating 124 as a single token, we need the permutations of 0123456789, of which there are 8!. Thus, $|A_{12} \cap A_{24}| = 8!$. The number of valid passwords is 10! $- 2 \times 9! + 8! = 2943360.

(b) Each element in $A$ can map to $B$, so the total number of functions is $n^m$. Let $B = \{b_1, b_2, \ldots, b_n\}$. Let $F_1$ be the functions which do not have $b_1$ in the range (so no element of $A$ maps to $b_1$); similarly define $F_2, \ldots, F_m$. A function is not onto if one of the $b_i$'s is not in the range. That is, the functions which are not onto are in $F_1 \cup F_2 \cup \cdots \cup F_m$. Let us compute the size of a $k$-way intersection $|F_{i_1} \cap F_{i_2} \cap \cdots \cap F_{i_k}|$ - the functions which do not use $b_{i_1}, b_{i_2}, \ldots, b_{i_k}$. The function can map each element of $A$ to $m - k$ of the $b_i$'s, so there are $(m-k)^n$ such functions. Since there are $\binom{n}{k}$ such k-way intersections, by inclusion-exclusion,

$$|F_1 \cup F_2 \cup \cdots \cup F_m| = \sum_{k=1}^{m} (-1)^{k+1} \binom{m}{k} (m-k)^n.$$  

The number of onto functions is $m^n - |F_1 \cup F_2 \cup \cdots \cup F_m|$, 

\[
\text{number of onto functions from } [n] \text{ to } [m] = m^n - \sum_{k=1}^{m} (-1)^{k+1} \binom{m}{k} (m-k)^n = \sum_{k=0}^{m} (-1)^k \binom{m}{k} (m-k)^n.
\]

Recall that in the solution to Exercise 13.3(c) we introduced $\{\binom{n}{m}\}$ (Stirling numbers of the second kind): the number of ways to distribute $n$ named objects (the elements of $A$) into $m$ unnamed (indistinguishible) bins (the elements of $B$) so that no bin is empty. Here, the bins are distinguishible, so multiplying by $m!$, the number of ways to label the bins with the labels $b_1, \ldots, b_m$, we have that $m! \{\binom{n}{m}\}$ is the number of ways to distribute the $n$ objects (A) into $m$ bins (B), each such way being an onto function from $A$ to $B$, so

\[
\text{number of onto functions from } [n] \text{ to } [m] = m! \{\binom{n}{m}\} = \sum_{k=0}^{m} (-1)^k \binom{m}{k} (m-k)^n.
\]

This gives us another formula for the Stirling number, $\{\binom{n}{m}\} = \frac{1}{m!} \sum_{k=0}^{m} (-1)^k \binom{m}{k} (m-k)^n$.

(c) If there were no upper bound constraints, we know how to compute the number of solutions:

\[
\text{number of solutions to } x_1 + x_2 + x_3 = 30 \text{ with } x_1, x_2, x_3 \geq 0 \text{ is } Q(30, 3) = \binom{32}{2} = 496.
\]

(We used $Q(n, k) = \binom{n+k-1}{k-1}$.) Let $S_i$ be the solutions which violate the upper bound constraint on $x_i$. So $S_1$ contains solutions to $x_1 + x_2 + x_3 = 30$ with $x_1 \geq 11$ and $x_2, x_3 \geq 0$. Similarly, define $S_2$, the solutions to $x_1 + x_2 + x_3 = 30$ with $x_2 \geq 16$ and $x_1, x_3 \geq 0$ and $S_3$, the solutions to $x_1 + x_2 + x_3 = 30$ with $x_3 \geq 21$ and $x_1, x_2 \geq 0$. The number of solutions which satisfy the upper bound constraints is $Q(30, 3) - |S_1 \cup S_2 \cup S_3|$. By inclusion-exclusion,

\[
|S_1 \cup S_2 \cup S_3| = |S_1| + |S_2| + |S_3| - |S_1 \cap S_2| - |S_1 \cap S_3| - |S_2 \cap S_3| + |S_1 \cap S_2 \cap S_3|.
\]

To get $|S_1|$, observe that solutions to $x_1 + x_2 + x_3 = 30$ with $x_1 \geq 11$ and $x_2, x_3 \geq 0$ are solutions to $x_1 + x_2 + x_3 = 19$ with $x_1, x_2, x_3 \geq 0$ after subtracting 11 from $x_1$ (and vice versa). Therefore $|S_1| = Q(19, 3)$. Similarly, $|S_2| = Q(14, 3)$ and $|S_3| = Q(9, 3)$. $S_1 \cap S_2$ contains solutions to $x_1 + x_2 + x_3 = 30$ with $x_1 \geq 11, x_2 \geq 16, x_3 \geq 0$. These solutions give solutions to $x_1 + x_2 + x_3 = 3$ with $x_1, x_2, x_3 \geq 0$ after subtracting 11 from $x_1$ and 16 from $x_2$, so $|S_1 \cap S_2| = Q(3, 3)$. Similarly, $|S_1 \cap S_3| = Q(-2, 3) = 0$ and $|S_2 \cap S_3| = Q(-7, 3) = 0$. Finally, $S_1 \cap S_2 \cap S_3$ contains solutions to $x_1 + x_2 + x_3 = 30$ with $x_1 \geq 11, x_2 \geq 16, x_3 \geq 21$.

480
which are solutions to $x_1 + x_2 + x_3 = -18$ with $x_1, x_2, x_3 \geq 0$ after subtracting 11 from $x_1$, 16 from $x_2$ and 21 from $x_3$. So, $|S_1 \cap S_2 \cap S_3| = Q(-18, 3) = 0$. Putting all this together,

$$\begin{align*}
|S_1 \cup S_2 \cup S_3| &= Q(19, 3) + Q(14, 3) + Q(9, 3) - Q(3, 3) - 0 - 0 + 0 \\
&= \binom{21}{2} + \binom{16}{2} + \binom{12}{2} - \binom{5}{2} \\
&= 375.
\end{align*}$$

Our answer is $496 - 375 = 121$ solutions.

(d) The numbers in $[n]$ which are not relatively prime to $n$ are divisible by $p_1$ or $p_2$ or . . . or $p_m$. Let $A_i$ be the numbers in $[n]$ that are divisible by $p_i$. Then the numbers in $[n]$ which are not relatively prime to $n$ are in $A_1 \cup \cdots \cup A_m$, and so

$$\varphi(n) = n - |A_1 \cup \cdots \cup A_m|.$$

To compute the size of the union, we need the sizes of $k$-way intersections $|A_{i_1} \cap \cdots \cap A_{i_k}|$, where $A_{i_1} \cap \cdots \cap A_{i_k}$ contains the numbers divisible by $p_{i_1}$ and $p_{i_2}$ and . . . $p_{i_k}$,

$$|A_{i_1} \cap \cdots \cap A_{i_k}| = \left\lceil \frac{n}{p_{i_1}p_{i_2}\cdots p_{i_k}} \right\rceil = \frac{n}{p_{i_1}p_{i_2}\cdots p_{i_k}},$$

where the last equality follows because $n$ is divisible by $p_{i_1}p_{i_2}\cdots p_{i_k}$. By inclusion-exclusion,

$$|A_1 \cup \cdots \cup A_m| = \sum_{k=1}^{m} (-1)^{k+1} \cdot \left( \text{sum over all } k\text{-way products of } n/p_{i_1}p_{i_2}\cdots p_{i_k} \right).$$

Therefore, Euler’s totient function is

$$\varphi(n) = n - \sum_{k=1}^{m} (-1)^{k+1} \cdot \left( \text{sum over all } k\text{-way products of } n/p_{i_1}p_{i_2}\cdots p_{i_k} \right)$$

$$= n \sum_{k=0}^{m} (-1)^{k} \cdot \left( \text{sum over all } k\text{-way products of } 1/p_{i_1}p_{i_2}\cdots p_{i_k} \right).$$

Let us compare this expression with the formula

$$\varphi(n) = n \prod_{j=1}^{m} \left( 1 - \frac{1}{p_j} \right) = n \left( 1 - \frac{1}{p_1} \right) \left( 1 - \frac{1}{p_2} \right) \cdots \left( 1 - \frac{1}{p_m} \right).$$

If you multiply out the RHS, you get $2^n$ terms. Each term is the product of $k$ reciprocal primes, $1/p_{i_1}p_{i_2}\cdots p_{i_k}$ with sign $(-1)^k$. Each term in this expansion matches one term in our inclusion-exclusion sum, and so the two expressions are equal.

Exercise 14.6.

(a) This is the same result as social twins, except with social enemies.

(b) You need to research two facts to solve this problem:

- Population of New York City: larger than 8 million.
- Human head-hairs: the estimate is at most 200,000. So let’s be safe and say 1 million.

Each person in New York City is a pigeon. The pigeonholes are 0, 1, . . . , 100. Each person is placed in the pigeonhole corresponding to the number of hairs on his/her head. Since there are more people than pigeonholes, at least two people occupy the same pigeonhole. Those two people have the same number of head-hairs.

(c) Define pigeonholes 0, 1, . . . , $k-1$. There are $k$ pigeonholes. The $k+1$ numbers are the pigeons. Place a number $x$ in pigeonhole $i$ if the remainder when $x$ is divided by $k$ is $i$. That is,

$$\text{pigeonhole}(x) = \text{rem}(x, k).$$

There are more numbers than pigeonholes, so at least two numbers $x_1$ and $x_2$ are in the same pigeonhole. That is, $\text{rem}(x_1, k) = \text{rem}(x_2, k)$, or $x_1 \equiv x_2 \pmod{k}$ and $k|x_1 - x_2$.

(d) There are 7 days of the week (pigeonholes). Place 8 guests (pigeons) in pigeonholes by the day of the week on which they were born, there will be at least two guests in the same pigeonhole. Even with infinitely many guests, none may be born on Monday. This is a very important aspect of the pigeonhole principle. You can guarantee two are born on the same day with 8 guests, but you don’t know which day that would be (it could be any).

(e) (i) Let $z_i$ be the number of pigeons in pigeonhole $i$, for $i = 1, \ldots, k$. Then $n = \sum_{i=1}^{k} z_i$.

Now suppose that no pigeonhole has at least $\lceil n/k \rceil$ pigeons, $z_i < \lceil n/k \rceil$. Since $\lceil n/k \rceil$
is an integer, it means that \( z_i \leq \lfloor n/k \rfloor - 1 \). It is easy to verify that \( \lfloor n/k \rfloor - 1 < n/k \) by separately considering the cases \( n/k \) is an integer and \( n/k \) is not an integer. Therefore,

\[
 n = \sum_{i=1}^{k} z_i \leq \sum_{i=1}^{k} (\lfloor n/k \rfloor - 1) < \sum_{i=1}^{k} n/k = n.
\]

This contradiction proves that \( z_i \geq \lfloor n/k \rfloor \) for at least one \( i \).

(ii) Certainly the longest non-decreasing subsequence ending at \( x_i \) contains \( x_i \), so \( 1 \leq \ell_i \). By assumption, \( \ell_i \leq n \). So, there are \( n \) possibilities for \( \ell_i \), \( 1, 2, \ldots, n \). Define the pigeonholes \( 1, 2, \ldots, n \). The numbers \( x_1, \ldots, x_{n+1} \) are the pigeons. Place the number \( x_i \) into the pigeonhole corresponding to \( \ell_i \). Since there are \( n^2+1 \) pigeons and \( n \) pigeonholes, there is at least one pigeonhole with at least \( \lceil (n^2+1)/n \rceil \) pigeons, by part (i). \( \lceil (n^2+1)/n \rceil = \lfloor n+1/n \rfloor = n+1 \). That is, there are at least \( n+1 \) of the \( \ell_i \) that are equal.

(iii) Suppose \( \ell_i = \ell_j \). Then we show that \( x_j < x_i \). Suppose, to the contrary, that \( x_j \geq x_i \). The longest non-decreasing sequence ending at \( x_i \) has length \( \ell_i \). Take this sequence and add \( x_j \) to the end: since \( x_j \geq x_i \), this is a non-decreasing sequence that ends at \( x_j \), so the longest non-decreasing sequence ending at \( x_j \) has length at least \( \ell_i+1 \) which contradicts the fact that \( \ell_j = \ell_i \). So \( x_j < x_i \).

We have proved that if \( \ell_{i_1} = \ell_{i_2} = \cdots \ell_{i_{n+1}} \), then \( x_{i_1} < x_{i_2} < \cdots < x_{i_{n+1}} \). That is, \( x_{i_1}, \ldots, x_{i_{n+1}} \) are non-increasing.

By (ii), there are \( n+1 \) of the \( \ell_i \) that are equal. That is, \( \ell_{i_1} = \cdots \ell_{i_{n+1}} \), which means \( x_{i_1}, \ldots, x_{i_{n+1}} \) are a non-increasing subsequence of length \( n+1 \), concluding the proof. This result is tight in that there are \( n^2 \) numbers for which there are non-increasing and non-decreasing sequences of length \( n \) but neither of length \( n+1 \). For example,

\[
2, 4, 1, 3 \quad (n = 2) \quad 3, 6, 9, 2, 5, 8, 1, 4, 7 \quad (n = 3)
\]

(f) There are 4 suits (pigeonholes). The cards are pigeons. Place a card into the pigeonhole corresponding to its suit. By part (e) sub-part (i), there are at least \( \lceil 17/4 \rceil = 5 \) cards in one pigeonhole (suit). That is, there must be a 5-card flush.

(g) The pigeons are the baskets. The pigeonholes are 0, 1, \ldots, 24. Place a basket into the pigeonhole corresponding to the number of apples in the basket. There are 51 baskets and 25 pigeonholes. By part (e) sub-part (i), there are at least \( \lceil 51/25 \rceil = 3 \) baskets in one of the pigeonholes. That is at least 3 baskets contain the same number of apples.

(h) Suppose none of the unit squares overlap. The area covered by the squares is 13. The area of a circle of radius 2 is \( A = 4\pi \approx 12.57 < 13 \). If the squares do not overlap, they cover an area larger than the circle, a contradiction. Therefore, the squares must overlap.

Chapter 15

Pop Quiz 15.1.

(a) The statement involves John Smith and liver disease. The crucial aspect about a probabilistic statement is to determine exactly what is the “source of the randomness”. Is it that 30% of the John Smiths will survive till seventy? Is it that 30% of the John Smiths who are diagnosed with liver disease will survive till seventy? Is it that 30% of people diagnosed with liver disease will survive till seventy (and this persons name being John Smith is incidental)? Based on our intuitive understanding of the context, most would agree that the name is incidental, and the probabilistic statement is being made about people diagnosed with liver disease.

So, my interpretation of this statement is that if you took all people diagnosed with liver disease, about 30% of them will survive till seventy.

(b) Among all packets sent out on the internet, approximately 0.01% are “dropped” and about 99.99% of them successfully reach their destination.

(c) Between now and your wedding in Washington, several “random things will occur”. Taken in sequence, some of these combinations of random things will allow me to come to the wedding (e.g. I finish my thesis ahead of schedule); and, in some cases, I won’t be able to come (e.g. one of my experiments fails and I need to redo it). Half of those combinations of random things will allow me to come and half won’t. This is similar to “The chance of rain tomorrow is 40%”. If we “re-live” the time between now and your wedding 100 times, on approximately 50 of those reincarnations I will be at your wedding.
Pop Quiz 15.2. Here is the outcome-tree with edge probabilities and outcome-probabilities.

Every edge probability is $\frac{1}{6}$ because at each vertex the die randomly rolls one of 6 possible values. The outcomes where you win are when the second roll is larger than the first. These outcomes are shaded and form the event of interest. Adding the outcome probabilities for this event,

$$P[\text{"Win"}] = \frac{1}{36} \times (5 + 4 + 3 + 2 + 1) = \frac{5}{12}.$$ 

The probability to win the dice game is less, by $\frac{1}{12}$ than the probability to win the coin game. If you played many games of each, you would win $8\frac{1}{12}$% more coin games, so you prefer the coin game.

We mention a more convenient representation for the outcomes of a pair of dice. Instead of using a cumbersome tree, we can use a two dimensional grid illustrated on the right. On the x-axis are rolls for die 1 and on the y-axis are rolls for die 2. The outcomes are pairs, one from the x-axis and one from the y-axis, and every pair has the same outcome probability $\frac{1}{36}$. This representation of the outcomes is much more compact than the tree. The event of interest is the same, and we have shaded the outcomes where you win. As above, there are 15 outcomes in the event of interest, so again, $P[\text{"Win"}] = \frac{15}{36} = \frac{5}{12}$.

The outcome-tree is the mechanism to obtain the outcomes and outcome-probabilities. Once you have the outcomes and outcome probabilities, you have no further need for the outcome tree. What matters are the outcomes, when defining the event of interest, and the outcome-probabilities when computing the probability of the event. Thus, it is often more convenient to represent the outcomes and their probabilities in a more compact way as we did here with a grid.

Pop Quiz 15.3. From partial outcome (1,2), the host, having to open an empty door, must open door 3. So, the host has only one option. From partial outcome (1,1), the host has a choice of doors because both are empty. The two edges correspond to the two choices.

Exercise 15.4.

(a) If you implement the algorithm and repeatedly run with $n = 120$, you might get different answers each time. So there is no “right” answer. We ran it once and got

<table>
<thead>
<tr>
<th>outcome</th>
<th>(3,2)</th>
<th>(2,3)</th>
<th>(1,2)</th>
<th>(1,3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>number of times</td>
<td>42</td>
<td>34</td>
<td>22</td>
<td>22</td>
</tr>
<tr>
<td>Prize Door</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>number of times</td>
<td>44</td>
<td>34</td>
<td>42</td>
<td></td>
</tr>
</tbody>
</table>

Each prize door occurs roughly $\frac{1}{3}$rd of the time. The prize has no “preference” for any door.

Half of the time when the prize door is 1, the host opens door 2, and the other half the time, he opens door 3. When the prize door is 1, the host has no “preference” for a door.

(b) If you switch, you win for the outcomes (3,2) and (2,3), which is 76 times.

(c) Switching is better: you win 76 times versus 44 times if you stay.
On the right we plot the win/loss ratio as you play more and more games. As the number of games played becomes larger, by switching, your win/loss ratio appears to “converge” to 2. You win about twice as often if you switch.

Pop Quiz 15.5.

(a) We use the 6-step method with the outcome-tree.

(i) Instead of the contestant choosing door 1, the contestant chooses any door he wishes.

Since we are not told the probabilities with which the contestant chooses each door, we denoted these (possibly different) probabilities by \( p_1, p_2, p_3 \), where \( p_1 + p_2 + p_3 = 1 \). By switching, the contestant wins for the outcomes in the event

\[ \mathcal{E} = \{(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 2, 1), (3, 1, 2)\} \]

The probability of winning by switching is \( \mathbb{P}[\mathcal{E}] \) which is given by

\[
\mathbb{P}[^{\mathcal{E}}] = \frac{1}{2}p_1 + \frac{1}{2}p_2 + \frac{1}{2}p_3 + \frac{1}{2}p_2 + \frac{1}{2}p_3 + \frac{1}{2}p_1
\]

\[
= \frac{2}{3} (p_1 + p_2 + p_3)
\]

\[
= \frac{2}{3}.
\]

The probability of winning by switching has not changed. This is to be expected, because all we are doing is relabeling the doors: the contestant’s door is relabeled as door 1. The other two doors are arbitrarily labeled 2 and 3, and the game has not changed.

(ii) The outcome-tree is the same; the edge-probabilities change.
### Exercise 15.6.

(a) In the table, we show the result of 10,000 games using the boxed Monte Carlo algorithm.

<table>
<thead>
<tr>
<th>Die Battle</th>
<th>A versus B</th>
<th>B versus C</th>
<th>A versus C</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result</td>
<td>A wins 5527 games</td>
<td>B wins 5531 games</td>
<td>C wins 5538 games</td>
</tr>
</tbody>
</table>

```
1: dieA=[2,2,6,6,7,7]; dieB=[1,1,5,5,9,9]; % Die faces
2: NumGames = 1000; % Number of games
3: Awins = 0;
4: for games = 1 to NumGames do
5:    a ← random value from dieA;
6:    b ← random value from dieB;
7:    if a > b then
8:        Awins ← Awins + 1; % Die A wins
9:    return Awins
```

(b) Since we are on a roll with Monte-Carlo, lets see what simulation gives sum of two rolls:

<table>
<thead>
<tr>
<th>A versus B</th>
<th>B versus C</th>
<th>A versus C</th>
</tr>
</thead>
<tbody>
<tr>
<td>B wins 5171 games</td>
<td>C wins 5172 games</td>
<td>A wins 5371 games</td>
</tr>
<tr>
<td>A wins 4616 games</td>
<td>B wins 4584 games</td>
<td>C Wins 4143 games</td>
</tr>
<tr>
<td>tie 213 games</td>
<td>tie 244 games</td>
<td>tie 486 games</td>
</tr>
</tbody>
</table>

The outcome-probabilities have not changed for the outcomes in which you win by
switching. The probability to win by switching is still $\frac{2}{3}$.

(b) We add a level in the outcome-tree for what the contestant does after the host opens a door.

The probability of the outcomes where the contestant wins are highlighted with gray shading.
Adding these outcome-probabilities gives $P[\text{ContestantWins}] = \frac{1}{12} + \frac{1}{12} + \frac{1}{6} + \frac{1}{6} = \frac{1}{2}$.
### Solutions to Quizes & Exercises

1: `dieA=[2,2,6,6,7,7]; dieB=[1,1,5,5,9,9];`  
2: `NumGames = 1000;`  
3: `Awins = 0; Bwins = 0;`  
4: ```python
for games = 1 to NumGames do
  a ← sum of two random values from dieA;
  b ← sum of two random values from dieB;
  if a > b then
    Awins ← Awins + 1;  % Die A wins
  else
    if b > a then
      Bwins ← Bwins + 1;  % Die B wins
    end if
  end if
end for
```
5: `return Awins, Bwins`  

The winners of each battle are different. Though A dominates B in one roll, B dominates when you take the sum of two rolls. Very strange. Let’s now do the outcome-tree analysis.

An outcome specifies the value of 4 rolls. For example, in the battle of A versus B, the outcome `(2,2)(1,1)` stands for die A rolling 2 then 2, and die B rolling 1 then 1. Here are the die A outcomes (we ordered them by sum):

<table>
<thead>
<tr>
<th>Outcome</th>
<th>(2,2)</th>
<th>(2,6)</th>
<th>(6,2)</th>
<th>(2,7)</th>
<th>(7,2)</th>
<th>(6,6)</th>
<th>(6,7)</th>
<th>(7,6)</th>
<th>(7,7)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum</td>
<td>4</td>
<td>8</td>
<td>8</td>
<td>9</td>
<td>9</td>
<td>12</td>
<td>13</td>
<td>13</td>
<td>14</td>
</tr>
</tbody>
</table>

Here are the die B outcomes:

<table>
<thead>
<tr>
<th>Outcome</th>
<th>(1,1)</th>
<th>(1,5)</th>
<th>(5,1)</th>
<th>(1,9)</th>
<th>(9,1)</th>
<th>(5,5)</th>
<th>(5,9)</th>
<th>(9,5)</th>
<th>(9,9)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum</td>
<td>2</td>
<td>6</td>
<td>6</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>14</td>
<td>14</td>
<td>18</td>
</tr>
</tbody>
</table>

Here are the die C outcomes:

<table>
<thead>
<tr>
<th>Outcome</th>
<th>(3,3)</th>
<th>(3,4)</th>
<th>(4,3)</th>
<th>(4,4)</th>
<th>(3,8)</th>
<th>(8,3)</th>
<th>(4,8)</th>
<th>(8,4)</th>
<th>(8,8)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>8</td>
<td>11</td>
<td>11</td>
<td>12</td>
<td>12</td>
<td>16</td>
</tr>
</tbody>
</table>

There are 9 outcomes for each die (2 rolls) and so in a battle, there are 81 possible outcomes (product rule). All the outcomes have the same probability `1/81`. 

#### A versus B

Let us count the outcomes where B beats A. If A is (2,2), 8 of B’s outcomes beat the sum 4. In this way, for each outcome of A we count the outcomes of B which beat A, and then add them up (sum rule) to get the number of outcomes where B beats A:

Number of outcomes where B beats A = `8 + 6 + 6 + 6 = 42`. 

Similarly, the number of outcomes where A beats B and A ties B are:

- Number of outcomes where A beats B = `9 + 8 + 8 + 4 + 4 + 0 + 0 + 0 + 0 = 37`;
- Number of outcomes where A ties B = `0 + 0 + 0 + 0 + 0 + 0 + 0 + 0 + 2 = 2`.

Therefore, B wins over A because 

\[ P["A beats B"] = \frac{37}{81}; \quad P["B beats A"] = \frac{42}{81}; \quad P["A ties B"] = \frac{2}{81}. \]

#### B versus C

We perform the same analysis:

- Number of outcomes where C beats B = `9 + 8 + 8 + 5 + 5 + 1 + 1 + 0 + 0 = 42`;
- Number of outcomes where B beats C = `6 + 6 + 6 + 3 + 3 + 3 + 3 + 1 = 37`;
- Number of outcomes where B ties C = `0 + 1 + 1 + 0 + 0 + 0 + 0 + 0 + 0 = 2`.

Therefore, C wins over B because 

\[ P["B beats C"] = \frac{37}{81}; \quad P["C beats B"] = \frac{42}{81}; \quad P["B ties C"] = \frac{2}{81}. \]

#### A versus C

We perform the same analysis:

- Number of outcomes where C beats A = `9 + 5 + 5 + 5 + 5 + 1 + 1 + 1 + 1 = 33`;
- Number of outcomes where A beats C = `8 + 8 + 8 + 6 + 4 + 4 + 3 + 3 + 0 = 44`;
- Number of outcomes where A ties C = `0 + 1 + 1 + 0 + 0 + 2 + 0 + 0 + 0 = 4`.

Therefore, A wins over C because 

\[ P["A beats C"] = \frac{44}{81}; \quad P["C beats A"] = \frac{33}{81}; \quad P["A ties C"] = \frac{4}{81}. \]

(c) Now, for die A, `P(\(\bigcirc\)) = \frac{1}{2}`, `P(\(\bigotimes\)) = \frac{1}{5}` and `P(\(\square\)) = \frac{1}{5}`. Only battles involving A change.
Trick question. Given only the information in the question all we can do is give bounds on Exercise 15.8. Remember always that an event is a set of outcomes. We assume that students have only one color hair (hair cannot be both black and blonde).

(a) The outcomes are each student. Each student is equally likely.

(b) The event contains all students with black or blonde hair (80% of the students):

\[ P(\text{either black hair or blonde hair}) = 0.8. \]

(c) The event contains the 20% of students in the complement of the previous event:

\[ P(\text{neither black hair nor blonde hair}) = 0.2 \]

(d) Trick question. Given only the information in the question all we can do is give bounds on the probability. We want to know \(|\{\text{Black hair}\} \cup \{\text{Brown eyes}\}|\). This size is at least 60% of the students (the brown eyed ones) and at most all the students, so

\[ 0.6 \leq P(\text{either black hair or brown eyes}) \leq 1. \]

If we assume proportionality, then \(0.6 \times 50\% = 30\%\) of black haired students also have brown eyes. Then, by inclusion-exclusion,

\[ |\{\text{Black hair}\} \cup \{\text{Brown eyes}\}| = 50\% + 60\% - 30\% = 80\%, \]

and therefore, assuming proportionality,

\[ P(\text{either black hair or brown eyes}) = 0.8. \]

(e) Trick question. Again, we can only give bounds. If every black-haired student also has brown eyes, then the intersection can be as large as 50%. If all the other 50% of students have brown eyes, then the intersection is as small as 10%. Therefore,

\[ 0.1 \leq P(\text{black hair and brown eyes}) \leq 0.5. \]

If we assume proportionality, then \(P(\text{black hair and brown eyes}) = 0.3. \)

Pop Quiz 15.7. See the outcome-tree and outcome-probabilities in Exercise 15.6(c).

Exercise 15.8. Remember always that an event is a set of outcomes. We assume that students have only one color hair (hair cannot be both black and blonde).

(a) The outcomes are each student. Each student is equally likely.

(b) The event contains all students with black or blonde hair (80% of the students):

\[ P(\text{either black hair or blonde hair}) = 0.8. \]

(c) The event contains the 20% of students in the complement of the previous event:

\[ P(\text{neither black hair nor blonde hair}) = 0.2 \]

(d) Trick question. Given only the information in the question all we can do is give bounds on the probability. We want to know \(|\{\text{Black hair}\} \cup \{\text{Brown eyes}\}|\). This size is at least 60% of the students (the brown eyed ones) and at most all the students, so

\[ 0.6 \leq P(\text{either black hair or brown eyes}) \leq 1. \]

If we assume proportionality, then \(0.6 \times 50\% = 30\%\) of black haired students also have brown eyes. Then, by inclusion-exclusion,

\[ |\{\text{Black hair}\} \cup \{\text{Brown eyes}\}| = 50\% + 60\% - 30\% = 80\%, \]

and therefore, assuming proportionality,

\[ P(\text{either black hair or brown eyes}) = 0.8. \]

(e) Trick question. Again, we can only give bounds. If every black-haired student also has brown eyes, then the intersection can be as large as 50%. If all the other 50% of students have brown eyes, then the intersection is as small as 10%. Therefore,

\[ 0.1 \leq P(\text{black hair and brown eyes}) \leq 0.5. \]

If we assume proportionality, then \(P(\text{black hair and brown eyes}) = 0.3. \)

Exercise 15.9.

(a) We are told \(\mathcal{E}_1 \cap \mathcal{E}_2 = \emptyset\).

\[ P(\mathcal{E}_1 \cup \mathcal{E}_2) = \sum_{\omega \in \mathcal{E}_1 \cup \mathcal{E}_2} P(\omega) \overset{(*)}{=} \sum_{\omega \in \mathcal{E}_1} P(\omega) + \sum_{\omega \in \mathcal{E}_2} P(\omega) = P(\mathcal{E}_1) + P(\mathcal{E}_2). \]
(a) There are 36 outcomes in the uniform probability space. The outcomes are the elements of the sample space.

(b) \( E \) and \( \overline{E} \) are disjoint, so \( P[E \cap \overline{E}] = P[E] + P[\overline{E}] \). Since \( E \cap \overline{E} = \emptyset \) and \( P[\emptyset] = 1 \), we have that \( 1 = P[\Omega] = P[E \cap \overline{E}] = P[E] + P[\overline{E}] \).

(c) Consider \( Q \), alternatively, observe that \( q \rightarrow p \) means that whenever \( p \) is \( T \), \( q \) is also \( T \). That is, \( \omega \in E_q \rightarrow \omega \in E_p \), or \( E_p \subseteq E_q \). Since \( P(\cdot) \) is non-negative, the sum over outcomes in \( E_q \) includes the sum over outcomes in \( E_p \) plus possibly some other outcome-probabilities. That is, \( P[E_p] \leq P[E_q] \), which means \( P["p being true"] \leq P["q being true"] \).

(f) This follows from (e) because \( p \rightarrow p \land q \). Alternatively, observe that \( E_1 \cap E_2 \subseteq E_1 \).

**Exercise 15.10.**

(a) There are 36 outcomes in this uniform probability space (see Pop Quiz 15.2). The outcomes where the sum is 9 are \( \{\text{GGG} \oplus \text{GBB} \oplus \text{GBB} \oplus \text{BBB} \oplus \text{GGG} \oplus \text{GGG} \} \).

\( P[\text{"Sum is } 9"] = \frac{\# \text{ outcomes with sum } 9}{|\Omega|} = \frac{6}{36} = \frac{1}{6} \).

(b) You are not going to be able to draw the outcome tree here. To get to an outcome, e.g. \( \text{TTTTTTT} \) (ten tails in a row), you multiply 10 edge probabilities which are all \( \frac{1}{2} \). This is the same for every outcome, so we have a uniform probability space with \(|\Omega| = 2^{10}\) and \( P = 2^{-10} \). The number of sequences with 4 heads is \( \binom{10}{4} \). Therefore, \( P[\text{"4 heads"]} = \frac{\# \text{ outcomes with 4 heads}}{|\Omega|} = \frac{\binom{10}{4}}{2^{10}} \approx 0.2051 \).

(c) There are 3 choices for each roll: \{\text{GGG}, \text{GBB}, \text{BBB}\}. So, each edge probability is \( \frac{1}{3} \), giving a uniform probability space with \( 3^{10} \) possible outcomes.

(i) Choose 4 sevens in \( \binom{10}{4} \) ways; the remaining 6 rolls can be chosen in 2 ways each for \( 2^6 \) ways (product rule). So, there are \( \binom{10}{4} \times 2^6 \) outcomes (product rule again). Therefore, \( P[\text{4 sevens}] = \frac{\# \text{ outcomes with 4 sevens}}{|\Omega|} = \frac{\binom{10}{4} \times 2^6}{3^{10}} \approx 0.2276 \).

(ii) Choose the 4 sevens in \( \binom{10}{4} \) ways; of the remaining six rolls, choose 3 sixes in \( \binom{6}{3} \) ways. By the product rule, there are \( \binom{10}{4} \times \binom{6}{3} = 10!/4!3!3! \) outcomes. Therefore, \( P[\text{4 sevens and 3 sixes}] = \frac{\# \text{ outcomes with 4 sevens and 3 sixes}}{|\Omega|} = \frac{\binom{10}{4} \binom{6}{3}}{3^{10}} \approx 0.0711 \).

(iii) There are \( \binom{10}{4} \times 2^6 \) outcomes with 4 sevens; there are \( \binom{10}{4} \times 2^7 \) outcomes with 3 sevens; there are \( \binom{10}{4} \times \binom{6}{3} \) outcomes with 4 sevens and 3 sixes. By inclusion-exclusion, there are \( \binom{10}{4} \times 2^6 + \binom{10}{4} \times 2^7 - \binom{10}{4} \times \binom{6}{3} = 24,600 \) outcomes with 4 sevens or 3 sixes. So, \( P[\text{4 sevens or 3 sixes}] = \frac{\# \text{ outcomes with 4 sevens or 3 sixes}}{|\Omega|} = \frac{24600}{3^{10}} \approx 0.4166 \).
(d) In Exercise 13.13 we computed the number of Two-pair hands (123,552) and the number of Three-of-a-kind hands (54,912). So
\[
P[\text{"Two-pair"}] = \frac{123,552}{\binom{52}{13}} \approx 0.0475
\]
\[
P[\text{"Three-of-a kind"}] = \frac{54,912}{\binom{52}{13}} \approx 0.0211
\]
Three-of-a-kind should win because it is rarer.

(e) We have a uniform probability space in which each of the \(\binom{52}{13}\) possible 13-card hands is equally likely. A hand has no Ace means the 13 cards are selected from 48 cards. This can be done in \(\binom{48}{13}\) ways. So the probability to not have any Aces is \(\frac{\binom{48}{13}}{\binom{52}{13}}\). The complementary event is that the hand has an Ace, therefore
\[
P[\text{"Ace"}] = 1 - \frac{\binom{48}{13}}{\binom{52}{13}} = 1 - \frac{39 \times 38 \times 37 \times 36}{52 \times 51 \times 50 \times 49} \approx 0.6962.
\]

Exercise 15.11.

(a) The possible outcomes are all sequences of heads and tails which end in either HHT or THH and there is no earlier occurrence of HHT or THH. So, HHT (you win), HHIHT (you win), THH (friend wins), TTHTHH (friend wins) are possible outcomes, but TTHHT is not a possible outcome because though the sequence ends in HHT, there is an earlier occurrence of THH. The outcome space is infinite. It is quite a complicated outcome space. We cannot simply list out the possible outcomes with their probabilities. We show the how the game plays out as an outcome-tree.

```
Outcome  Probability  Winner
HH        \frac{1}{4}    You
HT        \frac{1}{4}    Friend
T         \frac{1}{2}    Friend
```

This outcome-tree is not showing outcomes of the game, but outcomes of coin tosses. However, based on these outcomes of the coin tosses, we are reasoning about the game. If a tail appears, and the game is not over, then we know your friend wins. This is because for you to win, there must occur HH; at the first occurrence of HH, a THH has appeared and your friend has won. If HH appears, then you win, because for your friend to win, a T must appear and at the first appearance of T, you have won.

You win only if the coin tosses start out HH, that is \(P[\text{"you win"] = \frac{1}{4}\).

(b) We use a similar reasoning here using outcomes of the coin tosses.

```
Outcome  Probability  Winner
HH        \frac{1}{4}    You
HTH       \frac{1}{8}    Friend
HTT       \frac{1}{8}    Restart
T         \frac{1}{2}    Restart
```

Let us explain the outcomes of the coin toss. As in (a), if the coin starts with HH, then you win, because for your friend to win, a tail must be tossed, but at the first toss of a tail, you win. If the coin tosses HTH, then your friend wins and the game ends. A winning sequence for either you or your friend must start with H. So, if the coin starts with HTT or T, both of you are waiting for an H to start a possible winning sequence. This is the same situation as at the beginning of the game where you are both waiting for H. So the game effectively restarts.

You win \((y)\) with probability \(\frac{1}{4}\); your friend wins \((r)\) with probability \(\frac{1}{8}\); or, the game “restarts” \((r)\) with probability \(\frac{1}{8}\).

So the outcomes of the game are of the form \(r^iy\) (i restarts followed by you winning), or \(r^i\) (i restarts followed by your friend winning), where \(i \geq 0\).

<table>
<thead>
<tr>
<th>Outcome</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>(r^i)</td>
<td>(\frac{1}{8} \times \left(\frac{5}{8}\right)^i)</td>
</tr>
<tr>
<td>(r^i)</td>
<td>(\frac{1}{8} \times \left(\frac{5}{8}\right)^i)</td>
</tr>
</tbody>
</table>
The outcomes where you win are H**Y, so the probability you win is
\[
P[\text{"you win"}] = \frac{1}{4} \left[ 1 + \left( \frac{5}{8} \right) + \left( \frac{5}{8} \right)^2 + \cdots \right] = \frac{1}{4} \times \frac{1}{1 - \frac{5}{8}} = \frac{2}{3}.
\]

(c) We use a similar reasoning here using outcomes of the coin tosses.

<table>
<thead>
<tr>
<th>Outcome</th>
<th>Probability</th>
<th>Winner</th>
</tr>
</thead>
<tbody>
<tr>
<td>HH</td>
<td>$\frac{1}{4}$</td>
<td>You</td>
</tr>
<tr>
<td>HT</td>
<td>$\frac{1}{4}$</td>
<td>?</td>
</tr>
<tr>
<td>THH</td>
<td>$\frac{1}{8}$</td>
<td>You</td>
</tr>
<tr>
<td>THT</td>
<td>$\frac{1}{8}$</td>
<td>Friend</td>
</tr>
<tr>
<td>TT</td>
<td>$\frac{1}{4}$</td>
<td>?</td>
</tr>
</tbody>
</table>

As above, for HH, you win at the first toss of tails. A similar reasoning applies to THH and THT is the winning sequence for your friend. The interesting cases are the questions marks when a tail is tossed, T or HT. Now, any number of tails can follow. At the first H, you win if the next toss is H and your friend wins if the next toss is T. Here are the outcomes:

<table>
<thead>
<tr>
<th>Outcome</th>
<th>Probability</th>
<th>Winner</th>
</tr>
</thead>
<tbody>
<tr>
<td>HH</td>
<td>$\frac{1}{4}$</td>
<td>you</td>
</tr>
<tr>
<td>HT</td>
<td>$\frac{5}{8} \times \left( \frac{1}{2} \right)^i$</td>
<td>you</td>
</tr>
<tr>
<td>HH</td>
<td>$\frac{1}{8} \times \left( \frac{1}{2} \right)^i$</td>
<td>friend</td>
</tr>
<tr>
<td>THT</td>
<td>$\frac{1}{4} \times \left( \frac{1}{2} \right)^i$</td>
<td>you</td>
</tr>
<tr>
<td>THT</td>
<td>$\frac{1}{4} \times \left( \frac{1}{2} \right)^i$</td>
<td>friend</td>
</tr>
</tbody>
</table>

In the table above, $i \geq 1$. The probability that you win is the sum of the probabilities for the outcomes HH, HT**HH and T**HH, for $i \geq 1$:
\[
P[\text{"you win"}] = \frac{1}{4} + \frac{1}{8} \sum_{i=1}^{\infty} \left( \frac{1}{2} \right)^i + \frac{1}{4} \sum_{i=1}^{\infty} \left( \frac{1}{2} \right)^i = \frac{1}{4} + \frac{1}{8} + \frac{1}{4} = \frac{5}{8}.
\]

(d) Monte Carlo is always a useful tool for checking the result of a probability analysis. Here is a simulation to compute the probability that you win given your string and your friend’s string.

```matlab
1: you = [1, 1, 0]; friend = [0, 1, 1]; % H=1, T=0
2: NumGames = 10000; % Number of games
3: Pwin = 0; % Probability you win
4: for games = 1 to NumGames do
5:   x ← random binary vector of length 3;
6:   while TRUE do
7:     if isequal(x, you) then
8:       win = 1; break out of while;
9:     else if isequal(x, friend) then
10:    win = 0; break out of while;
11:   else
13:    Pwin ← Pwin + win - Pwin/games; % Update the probability
14:   return Pwin
```

Let us justify the update in step 13. Let $W$ be the number of wins up to that point; $games$ is the number of games played (including the current game) and $win$ is the outcome of the current game. The previous value of $P\text{win}$ is $W/(games - 1)$ (0 if $games = 1$), that is $W = P\text{win}(games - 1)$. The updated value of $P\text{win}$ should be $(W + win)/games$
\[
P\text{win} ← \frac{W + win}{games} = \frac{P\text{win}(games - 1) + win}{games} = P\text{win} + \frac{win - P\text{win}}{games}.
\]

The results of the simulation for $10^4$, $10^5$ and $10^6$ games are shown below.
### 30. Solutions to Quizes & Exercises

#### Number of games

<table>
<thead>
<tr>
<th>Number of games</th>
<th>10,000</th>
<th>100,000</th>
<th>1,000,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probability you win (a)</td>
<td>0.2477</td>
<td>0.25</td>
<td>0.25</td>
</tr>
<tr>
<td>Probability you win (b)</td>
<td>0.6626</td>
<td>0.666</td>
<td>0.667</td>
</tr>
<tr>
<td>Probability you win (c)</td>
<td>0.6294</td>
<td>0.6235</td>
<td>0.6255</td>
</tr>
</tbody>
</table>

Note that the frequency (probability from the simulation) gets closer to the computed probability as you play more games.

(e) Once again we illustrate the power of simulation. By symmetry, we may assume your friend chooses a sequence that starts with H. For each of the 4 sequences your friend chooses, we can evaluate your 8 possible sequences and pick the best. Here are the simulation results.

<table>
<thead>
<tr>
<th>friend</th>
<th>your best choice</th>
<th>$P[&quot;your best wins&quot;]$</th>
<th>$P[&quot;your 2nd-best wins&quot;]$</th>
</tr>
</thead>
<tbody>
<tr>
<td>(i)</td>
<td>HHH</td>
<td>0.8747</td>
<td>0.7005</td>
</tr>
<tr>
<td>(ii)</td>
<td>HHT</td>
<td>0.7500</td>
<td>0.5000</td>
</tr>
<tr>
<td>(iii)</td>
<td>HTH</td>
<td>0.6668</td>
<td>0.6253</td>
</tr>
<tr>
<td>(iv)</td>
<td>HTT</td>
<td>0.6658</td>
<td>0.5004</td>
</tr>
</tbody>
</table>

Let us analyze each of the 4 cases for your friends choice of string.

(i) If a T is tossed, you win at the first arrival of HH. So your friend can only win if the game specifically starts HHH, which has probability $\frac{1}{8}$.

$$P["THH beats HHH"] = \frac{7}{8}.$$ You cannot do better with any other sequence (you must lose if the game starts HHH).

(ii) If the game starts HH (probability $\frac{1}{4}$), your friend wins at the first arrival of T. If the game starts any other way, you win at the first arrival of HH. So,

$$P["THH beats HHT"] = \frac{3}{4}.$$ (iii) We analyzed this in part (b).

$$P["HHT beats HTH"] = \frac{2}{3}.$$ (iv) Here are relevant outcomes of the coin tosses.

<table>
<thead>
<tr>
<th>Outcome</th>
<th>Probability</th>
<th>Winner</th>
</tr>
</thead>
<tbody>
<tr>
<td>HH</td>
<td>$\frac{1}{4}$</td>
<td>You</td>
</tr>
<tr>
<td>HTH</td>
<td>$\frac{1}{8}$</td>
<td>?</td>
</tr>
<tr>
<td>HTT</td>
<td>$\frac{1}{8}$</td>
<td>Friend</td>
</tr>
<tr>
<td>T</td>
<td>$\frac{1}{2}$</td>
<td>Restart</td>
</tr>
</tbody>
</table>

The outcome with the question mark is equivalent to restarting from the shaded H vertex. We conclude that the outcomes where you win start with any number of T’s followed by any number of HT’s followed by HH. That is, the outcomes where you win are $T^i HT^j HH$, having probability $(\frac{1}{2})^i \times (\frac{1}{4})^j \times \frac{1}{4}$. Therefore,

$$P["HHT beats HTT"] = \frac{1}{4} \sum_{i=0}^{\infty} \sum_{j=0}^{\infty} (\frac{1}{4})^i (\frac{1}{4})^j = \frac{1}{4} \times 2 \times \frac{1}{1 - \frac{1}{4}} = \frac{2}{3}.$$  

#### Chapter 16

**Pop Quiz 16.1.**

(a) Barring the unforeseen Humans will be here, $P["There is a living Human tomorrow"] \approx 1$.

(b) Similarly, $P["Sun does not rise tomorrow"] \approx 1$.

(c) The new information that is given (the Sun does not rise tomorrow) means that some catastrophe has indeed occurred. Humans are likely wiped out,

$$P["There is a living Human tomorrow \mid Sun does not rise tomorrow"] \approx 0.$$ The new information significantly changes the probability of Humans being around tomorrow.

**Exercise 16.2.**

(a) $P[CS \mid MATH] = \frac{P[CS \cap MATH]}{P[MATH]} = \frac{0.016}{0.02} = 0.8$; (In general, $P[A \mid B] \neq P[B \mid A]$.)

$$P[MATH \mid CS] = \frac{P[CS \cap MATH]}{P[CS]} = \frac{0.016}{0.2} = 0.08.$$
(b) (i) \(P[A|A] = P[A \cap A]/P[A] = 1.\)
(ii) \(P[A|A \cap B] = P[A \cap (A \cap B)]/P[A \cap B] = P[A \cap B]/P[A \cap B] = 1.\)
(iii) \(P[A \cap B] = P[(A \cap B) \cup B]/P[B] = P[A \cap B]/P[B] = P[A|B].\)
(iv) \(P[A \cup B] = P[(A \cup B) \cap B]/P[B] = P[B]/P[B] = 1.\)
(v) \(P[A \cup B] = P[A \cap (A \cup B)]/P[A \cup B] = P[A]/P[A \cup B].\)

(c) (i) \(P_B(\omega) = P[\{\omega\} \cap B]/P[B].\)
If \(\omega \notin B, P[\{\omega\} \cap B] = P[\emptyset] = 0;\) otherwise, \(P[\{\omega\} \cap B] = P[\{\omega\}] = P(\omega).\)
(ii) First, note that \(P[B] = \sum_{\omega \in B} P(\omega).\)
\[
\sum_{w \in B} P_B(\omega) = \sum_{w \in B} P_B(\omega) + \sum_{w \notin B} P_B(\omega) = \sum_{w \in B} \frac{P(\omega)}{P[B]} + \sum_{w \notin B} \frac{P(\omega)}{P[B]} = 1
\]

Pop Quiz 16.3.
(a) The outcomes are \((i,j),\) where \(i\) is the first roll and \(j\) is the second roll. So, \(1 \leq i, j \leq 6.\) This
is a uniform probability space so \(P(i,j) = 1/(\text{number of outcomes}) = \frac{1}{36}.\)
(b) The number of outcomes in the event is 3 so \(P(\text{Sum is 10}) = \frac{3}{36} = \frac{1}{12}.\)
(c) The number of outcomes in the event is 9 so \(P(\text{Both are Odd}) = \frac{9}{36} = \frac{1}{4}.\)
(d) Only 1 outcome has both dice odd and a sum 10, so \(P(\text{([Sum is 10] AND (Both are Odd)]}) = \frac{1}{36} = \frac{1}{36}.\)
(e) \(P(\text{Sum is 10 | Both are Odd}) = \frac{P(\text{([Sum is 10] AND (Both are Odd)])}}{P(\text{Both are Odd})} = \frac{\frac{1}{36}}{\frac{1}{4}} = \frac{1}{9}.\)
(f) \(P(\text{Both are Odd | Sum is 10}) = \frac{P(\text{([Sum is 10] AND (Both are Odd)])}}{P(\text{Sum is 10})} = \frac{\frac{1}{36}}{\frac{1}{12}} = \frac{1}{3}.

Exercise 16.4.
(a) The analysis mimics the example before Exercise 16.4 on page 234, with \(P(1,2) = 0\) and \(P(1,3) = \frac{1}{3}.\) We give the intuition. Since Monty will always open door 3 if it is available, it must be that door 3 is not available, so you win with probability 1 by switching.

(b) The outcome tree is shown on the left. We want \(P(\text{Win by Switching} | \text{Door 2 Opened}).\)

Exercise 16.5. On the left is the outcome-tree from which we can obtain the probabilities.

Exercise 16.6.
(a) Yes. You can only estimate $P[\text{student likes the course} \mid \text{student rated course}]$. Students who rate the course is not a random sampling of students. In general, any surveys of this type suffer from sampling bias. This type of sampling bias is sometimes called non-response bias. Those who do not respond tend to be a particular type of person, not a random sample.

(b) Wald first surmised that the hits taken on planes should be somewhat random (shot accuracy is not high enough in an aerial battle to target specific parts of a plane). So there should be just as many hits on the tail and nose as main body. So the returning war-planes are not indicative of where the planes are getting hit; there are indicative of which planes survive given they are hit. He concluded that

$$P[\text{survive} \mid \text{hit on mid-body}] \gg P[\text{survive} \mid \text{hit on nose or tail}].$$

Therefore, the nose and tail are the regions that needed fortification.

**Pop Quiz 16.7.** You are not old enough to have any profession but student.

**Pop Quiz 16.8.** We require $P[A \cap B] / P[B] = P[A \cap B] / P[A]$ (assuming $P[A], P[B] > 0$), or $P[A \cap B](P[A] - P[B]) = 0$. Either $A$ and $B$ must be disjoint so that $P[A \cap B] = 0$, or $P[A] = P[B]$.

**Exercise 16.9.**

(a) By (16.1) on page 238, $P[A_1 \cap (A_2 \cap A_3)] = P[A_1 \mid A_2 \cap A_3] \times P[A_2 \cap A_3]$. Applying (16.1) again,

$$P[A_1 \cap A_2 \cap A_3] = P[A_1 \mid A_2 \cap A_3] \times P[A_2 \mid A_3] \times P[A_3].$$

The general formula, which we encourage you to prove by induction, is

$$P[A_1 \cap A_2 \cap \cdots \cap A_n] = P[A_1 \mid A_2 \cap \cdots \cap A_n] \times P[A_2 \mid A_3 \cap \cdots \cap A_n] \times \cdots \times P[A_{n-1} \mid A_n] \times P[A_n].$$

(b) Let us consider an example path on the outcome-tree to a leaf. The outcome is $v_1v_2v_3v_4$:

[Diagram of an outcome tree with nodes $p_1, p_2, v_2, v_3, p_3, v_4, p_4$, showing the path $p_1 \rightarrow p_2 \rightarrow v_2 \rightarrow v_3 \rightarrow p_3 \rightarrow v_4 \rightarrow p_4$]

$p_1$ is the probability that $v_1$ occurs at the start, $p_1 = P[v_1]$. After $v_1$ occurs, $p_2$ is the probability that $v_2$ occurs, given $v_1$ has occurred, $p_2 = P[v_2 \mid v_1]$. Similarly, $p_3$ is the probability that $v_3$ occurs, given $v_1, v_2$ have occurred, $p_3 = P[v_3 \mid v_1$ AND $v_2]$; $p_4 = P[v_4 \mid v_1$ AND $v_2$ AND $v_3]$. By part (a),

$$P[v_3 \land v_2 \land v_1] = P[v_4 \mid v_1 \land v_2 \land v_3] \times P[v_3 \mid v_1 \land v_2] \times P[v_2 \mid v_1] \times P[v_1] = p_1p_2p_3p_4.$$ 

That is, the probability of the leaf outcome occurring is exactly the product of the edge probabilities leading to that leaf.

**Exercise 16.10.** By the definition of conditional probability, $P[A \mid B] = P[A \cap B] / P[B]$, and also, $P[A \cap B] = P[B \mid A] P[A]$. Therefore,

$$P_{\text{prob}}[A \mid B] = \frac{P[B \mid A] P[A]}{P[B]} = \frac{P[B \mid A] P[A]}{P[B] + P[B \mid \overline{A}] P[\overline{A}]}.$$ 

(The second equality is by the law of total probability, $P[B] = P[B \mid A] P[A] + P[B \mid \overline{A}] P[\overline{A}]$.)

(a) Using R for Republican and D for democrat,

$$P[\text{oppose taxes}] = P[\text{oppose taxes} \mid R] P[R] + P[\text{oppose taxes} \mid D] P[D] = 0.7 \times 0.4 + 0.5 \times 0.6 = 0.58.$$ 

(b) Using Bayes’ Theorem,

$$P[R \mid \text{oppose taxes}] = \frac{P[\text{oppose taxes} \mid R] P[R]}{P[\text{oppose taxes}]} = \frac{0.7 \times 0.4}{0.58} \approx 0.483.$$ 

**Exercise 16.11.** You win if two consecutive heads arrive before two consecutive tails.

(a) The outcomes where you win begin with (HT)* or T(HT)*, and end with HH.

<table>
<thead>
<tr>
<th>Winning outcomes</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>(HT)*HH</td>
<td>$(p(1-p))^i \times p^2$</td>
</tr>
<tr>
<td>T(HT)*HH</td>
<td>$(1-p) \times (p(1-p))^i \times p^2$</td>
</tr>
</tbody>
</table>

To get the probability of winning, we add these probabilities,

$$P[\text{win}] = p^2 \sum_{i=0}^{\infty} (p(1-p))^i + p^2(1-p) \sum_{i=0}^{\infty} (p(1-p))^i = \frac{p^2(2-p)}{1-p(1-p)}.$$
(b) By the law of total probability,
\[ P[\text{win}] = P[\text{win}|\text{H}] P[\text{H}] + P[\text{win}|\text{T}] P[\text{T}] \]
\[ = p P[\text{win}|\text{H}] + (1 - p) P[\text{win}|\text{T}] . \]
Let us compute \( P[\text{win}|\text{H}] \) and \( P[\text{win}|\text{T}] \) using total probability.
\[ P[\text{win}|\text{H}] = (1 - p) P[\text{win}|\text{T}] + p . \]
(If you get H you won, and if you get T, it is as if you started with T.) Similarly,
\[ P[\text{win}|\text{T}] = p P[\text{win}|\text{H}] . \]
(If you get T you lost, and if you get H, it is as if you started with H.) Using this expression
for \( P[\text{win}|\text{T}] \), we have: \( P[\text{win}|\text{H}] = p(1 - p) P[\text{win}|\text{H}] + p \). Solving for \( P[\text{win}|\text{H}] \):
\[ P[\text{win}|\text{H}] = \frac{p}{1 - p(1 - p)} \quad \text{and} \quad P[\text{win}|\text{T}] = \frac{p^2}{1 - p(1 - p)} . \]
Substituting back into \( P[\text{win}] \),
\[ P[\text{win}] = \frac{p^2}{1 - p(1 - p)} + \frac{p^2(1 - p)}{1 - p(1 - p)} = \frac{p^2(2 - p)}{1 - p(1 - p)} . \]

Chapter 17

Pop Quiz 17.1. This is a tricky question. The second toss is H with probability \( p \), and that
is independent of whether the first toss came up H or T. You can view this independence as
“full-knowledge” independence, or the “universe” view.
Suppose you must predict the second toss. Does knowing the first toss help you? Yes. From your
point of view, the two tosses are not independent. The first toss tells you about \( p \), which helps
you to predict the second toss. If the first toss is H, then you would guess that \( p > \frac{1}{2} \) and predict
the second toss as H. Imagine if the first 10 tosses are H. Now you most certainly would suspect
that \( p \approx 1 \) (biased coin) and predict the 11th toss as H. Compare: The sun has risen every day in
recorded history. Does that help you predict whether or not the sun will rise tomorrow?
If you fix \( p = \frac{1}{2} \), now you don’t care what the first toss was; \( p \) is what governs the second toss, so
the two tosses are independent. Your view becomes the full-knowledge (or universe) view.

Exercise 17.2.
(a) False. Consider two disjoint events, each having positive probability.
(b) False. Consider the rain and clouds example, or \( A = B \) with \( P[A] < 1 \).
(c) True. \( A \cap B \subseteq A \), therefore \( P[A \cap B] \leq P[A] \); similarly \( P[A \cap B] \leq P[B] \).

Exercise 17.3. We proved this formula in Exercise 16.10. Define the event
\[ A = \{ \text{Person is using a cellphone during a particular (fixed) weekday minute.} \} \]
Randomly pick a person in the USA. We want \( P[A] \). To use the Fermi method, we first break
down \( A \) into smaller events that are easier to analyze.
\[ A_1 = \{ \text{Has cellphone}\} ; \]
\[ A_2 = \{ \text{Uses cellphone in the particular minute}\} . \]
\( A \) occurs if \( A_1 \) and \( A_2 \) occur.
\[ P[A] = P[A_1 \cap A_2] = P[A_1] \times P[A_2 | A_1] \]
Typical surveys say that 9 in 10 adults (15-70 years old) have a cellphone and adults are about 50% of
the population. The typical cellphone plan is 1000 weekday minutes per month, which suggests
that people who have a cellphone use about 1000 weekday minutes. Assume that phone usage is
evenly spaced during the month and a weekday has 12 hours (8am-8pm) which is 720 minutes,
so 20 weekdays in a month equals 14400 weekday minutes. 1000 minutes are spread evenly over
these 14400 minutes. There are \( 14400 \) ways to pick 1000 of the 14400 minutes. If you do not use
a particular minute, there are only \( 14399 \) ways. The probability to not use a particular weekday
minute is \( \frac{14399}{14400} \). Since \( \binom{n}{k} = \frac{n!}{k!(n-k)!} \), \( \binom{14399}{14400} / \binom{14400}{14400} = 14400 - 14400 / 14400 = 1 \).
\[ P[\text{Has cellphone}] \approx \frac{9}{10} \times \frac{1}{2} = \frac{9}{20} . \]
\[ P[\text{Uses cellphone on the minute} | \text{Has cellphone}] \approx \frac{1000}{14400} \approx \frac{1}{15} . \]
Multiplying the probabilities, \( P[A] \approx \frac{2}{3} \times \frac{1}{3} = \frac{2}{9} \). There are 320 million people in the USA, so about 10 million are in the event \( A \) (for the USA).

**Pop Quiz 17.4.**

(a) Since the coins are independent, any pair of coins matches with probability \( \frac{1}{2} \). This is also verified by “brute force” because, for example, \( A_1 = \{ \text{HHH, HHT, TTH, TTT} \} \) contains 4 outcomes and each has probability \( \frac{1}{8} \) so \( P[A_1] = 4 \times \frac{1}{8} = \frac{1}{2} \).

(b) \( A_2 = \{ \text{HHH, THH, HTT, TTT} \} \), so \( A_1 \cap A_2 = \{ \text{HHH, TTT} \} \), and so \( P[A_1 \cap A_2] = 2 \times \frac{1}{8} = \frac{1}{4} \).

The other 2 cases are analogous.

(c) \( A_1 \cap A_2 \cap A_3 = \{ \text{HHH, TTT} \} \) (events \( A_1, A_2, A_3 \) simultaneously hold). So, \( P[A_1 \cap A_2 \cap A_3] = \frac{1}{8} \).

If coins 1 and 2 match and 2 and 3 match, then 1 and 3 must match, so any two pairs matching means the third matches as well.

**Pop Quiz 17.5.**

(a) \( A_1 \) (blue) contains 18 outcomes, as does \( A_2 \), so \( P[A_1] = P[A_2] = \frac{18}{32} = \frac{9}{16} \). \( A_3 \) contains 4 outcomes so \( P[A_3] = \frac{4}{32} = \frac{1}{8} \). The intersection of the three shaded regions contains just the one outcome \((3,6)\), and so \( P[A_1 \cap A_2 \cap A_3] = \frac{1}{32} \), which is the product of the three probabilities.

That is, we have 3-way independence.

(b) \( A_1 \cap A_2 \cap A_3 \) contains the one outcome \((3,6)\), so \( P[A_1 \cap A_2 \cap A_3] = \frac{1}{32} \) and \( P[A_1] \cdot P[A_2] = \frac{1}{4} \).

(c) \( A_2 \cap A_3 \) contains the three outcomes \{(3,6), (4,5), (5,4)\} so \( P[A_2 \cap A_3] = \frac{3}{32} = \frac{3}{16} \) and \( P[A_2] \cdot P[A_3] = \frac{1}{16} \).

(d) The events are not 2-way independent.

**Pop Quiz 17.7.**

(a) We use the definition of conditional probability.

\[
P[A_3 \mid A_1 \cap A_2] = \frac{P[A_1 \cap A_2 \cap A_3]}{P[A_1 \cap A_2]} = \frac{P[A_1] \cdot P[A_2] \cdot P[A_3]}{P[A_1] \cdot P[A_2]} = P[A_3].
\]

The penultimate step is by independence.

(b) If \( X = A_1, Y = A_2, Z = A_3 \) the probability of the intersection equals the product by independence. Suppose that exactly one of sets is the complement, for example \( Z = \overline{A_3} \). Then, by the law of total probability,

\[
P[A_1 \cap A_2 \cap \overline{A_3}] + P[A_1 \cap A_2 \cap A_3] = P[A_1 \cap A_2] \rightarrow P[A_1 \cap A_2 \cap \overline{A_3}] = P[A_1 \cap A_2] - P[A_1 \cap A_2 \cap A_3].
\]

Now we can use independence to obtain

\[
P[A_1 \cap A_2 \cap \overline{A_3}] = P[A_1] \cdot P[A_2] - P[A_1] \cdot P[A_2] \cdot P[A_3] = P[A_1] \cdot P[A_2] \cdot (1 - P[A_3]) = P[A_1] \cdot P[A_2] \cdot P[\overline{A_3}]
\]

Let us give the general case and prove it by induction. Suppose \( A_1, A_2, \ldots, A_n \) are independent. Let \( X_i = A_i \) or \( \overline{A_i} \). Then we want to show that

\[
P[X_1 \cap \cdots \cap X_n] = P[X_1] \times \cdots \times P[X_n].
\]

We prove a stronger claim by induction: the probability of any \( k \)-way intersection of the \( X \)'s equals the product of the \( k \) probabilities. The base case, \( k = 1 \), trivially holds. Suppose the claim holds for any \( k \)-way intersection and consider a \((k+1)\)-way intersection. We must prove:

\[
P[X_{i_1} \cap \cdots \cap X_{i_{k+1}}] = P[X_{i_1}] \times \cdots \times P[X_{i_{k+1}}].
\]

We use a second induction on the number of complements among these \( k+1 \) sets. When there are no complements, the claim follows by independence of the \( X_i \). Suppose this claim holds when there are \( \ell \) complements and consider the case where there are \( \ell + 1 \) complements.

Without loss of generality, we may assume that \( X_{i_{k+1}} = \overline{A_{i_{k+1}}} \). Then,

\[
P[X_{i_1} \cap \cdots \cap X_{i_k} \cap \overline{A_{i_{k+1}}}] = P[X_{i_1} \cap \cdots \cap X_{i_k}] - P[X_{i_1} \cap \cdots \cap X_{i_k} \cap A_{i_{k+1}}]
\]

\[
= P[X_{i_1}] \times \cdots \times P[X_{i_k}] - P[X_{i_1} \cap \cdots \cap X_{i_k} \cap A_{i_{k+1}}]
\]

\[
= P[X_{i_1}] \times \cdots \times P[X_{i_k}] \times \left(1 - P[A_{i_{k+1}}]\right)
\]

\[
= P[X_{i_1}] \times \cdots \times P[X_{i_k}] \times P[\overline{A_{i_{k+1}}}].
\]

(The first step follows because the claim holds for all \( k \)-way intersections. The second because the second term has \( \ell \) complements and the claim holds for \( \ell \) complements.)
Exercise 17.6. Given $E_1, E_2, \ldots, E_{k-1}$, none of $s_1, \ldots, s_N$ are born on day 1 to day $\text{mathk}-1$. Suppose $s_k$ is born on day $k$, then $s_{k+1}, \ldots, s_N$ ($N-k$ students) are all born on days $k+1$ to $B$ ($B-k$ of the $B-k+1$ days, given $E_1, E_2, \ldots, E_{k-1}$). By independence,

$$P[E_k | E_1 \cap E_2 \cap \cdots \cap E_{k-1}] = \left(\frac{B-k}{B-k+1}\right)^{N-k}.$$ 

Exercise 17.8.

(a) We want

$$P = \prod_{k=1}^{N} \left(\frac{B-k}{B-k+1}\right)^{N-k},$$

or equivalently, $\ln P = \sum_{k=1}^{N} (N-k) \ln \left(\frac{B-k}{B-k+1}\right)$. The sum can be evaluated, but the product is numerically unstable. Observe that

$$\left(\frac{B-k+1}{B-k}\right)^{N-k} = \left(1 + \frac{1}{B-k}\right)^{N-k} \approx e^{(N-k)/(B-k)},$$

because $(1 + \frac{1}{x})^x \approx e$ for large $x$. Using this approximation in $P$,

$$P \approx \prod_{k=1}^{N} e^{-\frac{N-k}{B-k}} = e^{-\sum_{k=1}^{N} \frac{N-k}{B-k}}.$$

We need to evaluate the sum in the exponent,

$$\sum_{k=1}^{N} \frac{N-k}{B-k} = \sum_{k=1}^{N} \frac{N-B+k-B}{B-k} = \sum_{k=1}^{N} \frac{N-B}{B-k} + 1 = N + \sum_{k=1}^{N} \frac{N-B}{B-k}.$$

Using the integration method, $\sum_{k=1}^{N} \frac{N-B}{B-k} \approx (N-B) \int_{1}^{N} \frac{1}{B-x} \, dx = (N-B) \ln \frac{B-1}{B-N}$. Therefore,

$$P \approx e^{(B-N) \ln \frac{B-1}{B-N}}.$$

When $B = 366$ and $N = 200$, $P \approx e^{-68.2}$. Using the exact sum for $\ln P$, $P = e^{-68.4}$.

(b) Trick question. With 367 people and only 366 birthdays, by pigeonhole, at least two people must share the same birthday so the probability of no social-twins is 0.

(c) We compute $P = \sum_{k=1}^{N} (N-k) \ln \left(\frac{B-k}{B-k+1}\right)$, from which we obtain $P$

The probability of no social twin is $1 - P$. We show a plot $1 - P$ versus $N$ in the figure to the right. The probability of a social-twin rapidly increases and first goes over 0.5 for $N = 23$. So in a party of just 23, the odds favor there being a social-twin. By $N = 60$, it is essentially guaranteed that you have a social-twin.

(d) (i) Repetition is allowed. Each student has $B$ choices, so there are $B^N$ possible sequences.

(ii) Each sequence is equally likely so each has probability $1/B^N$. 

496
(iii) The first \(k\) birthdays are chosen in \(B(B-1)\cdots(B-k+1) = B^k/(B-k)!\) ways. The remaining \(N-k\) students choose (with repetition) from the remaining \(B-k\) birthdays in \((B-k)^{N-k}\) ways. By the product rule, the number of sequences with no repetitions of the first \(k\) birthdays is
\[
\frac{B!}{(B-k)!} \times (B-k)^{N-k}.
\]
(iv) Multiply the number of allowed sequences in (iii) by their probability in (ii) to get
\[
\mathbb{P}[\text{no repetition of first } k \text{ birthdays}] = \frac{1}{B^N} \times \frac{B!}{(B-k)!} \times (B-k)^{N-k}.
\]
(v) We can cancel many terms in Equation (17.5) as follows:
\[
\frac{(B-1)^{N-1}}{B^{N-1}} \times \frac{(B-2)^{N-2}}{(B-1)^{N-2}} \times \frac{(B-3)^{N-3}}{(B-2)^{N-3}} \cdots \frac{(B-k+1)^{N-k+1}}{(B-k)^{N-k+1}} \times \frac{(B-k)^{N-k}}{(B-k+1)^{N-k}}
\]
We highlighted pairs of terms in the same color which simplify: the red terms simplify to \((B-1)\); the green to \((B-2)\); and so on. We get:
\[
= \frac{1}{B^N} \times \frac{B!}{(B-k)!} \times (B-k)^{N-k}.
\]
(vi) When \(k = N\), the result follows from
\[
\frac{B(B-1)(B-2)\cdots(B-(N-1))}{B^N} = \frac{B}{B} \times \frac{(B-1)}{B} \times \frac{(B-2)}{B} \cdots \frac{(B-(N-1))}{B},
\]
To derive this formula directly, observe that
\[
\mathbb{P}[s_1 \text{ does not match } s_1] = 1 - \frac{1}{B},
\]
\[
\mathbb{P}[s_3 \text{ does not match any of } s_1, s_2 \mid \text{ no match in } s_1, s_2] = 1 - \frac{2}{B},
\]
\[
\mathbb{P}[s_4 \text{ does not match any of } s_1, s_2, s_3 \mid \text{ no match in } s_1, s_2, s_3] = 1 - \frac{3}{B},
\]
\[
\vdots
\]
\[
\mathbb{P}[s_N \text{ does not match any of } s_1, \ldots, s_{N-1} \mid \text{ no match in } s_1, \ldots, s_{N-1}] = 1 - \frac{N-1}{B}.
\]
Multiplying these conditional probabilities,
\[
\mathbb{P}[\text{no match in } s_1, s_2, \ldots, s_N] = (1 - \frac{1}{B}) \times (1 - \frac{2}{B}) \times (1 - \frac{3}{B}) \times \cdots \times (1 - \frac{N-1}{B}).
\]
**Exercise 17.9.**
(a) \(B = 300\) and \(N = 100\), so we can use \(e^{-N(N-1)/B} \leq \mathbb{P}[\text{no collisions}] \leq e^{-N(N-1)/2B}\):
\[
e^{-33} \leq \mathbb{P}[\text{no collisions}] \leq e^{-16.5} \quad \text{(essentially 0)}.
\]
(i) There are no collisions if and only if every bin has at most one object, so:
\[
e^{-33} \leq \mathbb{P}[\text{every bin has at most one object}] \leq e^{-16.5}.
\]
(ii) Some bin has more than one object if and only if there is a collision which has probability
\[
1 - \mathbb{P}[\text{no collisions}].
\]
Therefore,
\[
1 - e^{-16.5} \leq \mathbb{P}[\text{some bin has more than one object}] \leq 1 - e^{-33}.
\]
(b) We want \(\mathbb{P}[\text{no collisions}] \geq 0.9\), so we set \(e^{-N(N-1)/B} \geq 0.9\) which gives \(B \geq \lfloor N(N-1)/\ln(1/0.9) \rfloor\),
or \(B = 93,964\). That is a pretty big table size for just 100 words.
(c) \(\mathbb{P}[\text{no collisions}] \geq e^{-N(N-1)/B}\). When \(B = N^{2+\epsilon}, N(N-1)/B = N^{-\epsilon} - 1/N^{1+\epsilon} \to 0\). Therefore \(e^{-N(N-1)/B} \to 1\). Probabilities are at most 1, so \(\mathbb{P}[\text{no collisions}] \to 1\).
(d) \(\mathbb{P}[\text{no collisions}] \leq e^{-N(N-1)/2B}\). When \(B = N^{2-\epsilon}, N(N-1)/2B = 1/2N^\epsilon - 1/2N^{\epsilon-1} \to \infty\) for
\[
1 > \epsilon > 0.
\]
Therefore \(e^{-N(N-1)/2B} \to 0\). Probabilities are at least 0, so \(\mathbb{P}[\text{no collisions}] \to 0\).
Pop Quiz 17.10. By interchanging home and the lockup, \( P[\text{home}] = \frac{1}{4} \). Alternatively, the successful step-sequences are L(RL)*L, having probability \( \frac{1}{2} \times (\frac{1}{2})^3 \times \frac{1}{2} \). Summing these probabilities,
\[
P[\text{home}] = \frac{1}{4} \sum_{i=0}^{\infty} \left( \frac{1}{4} \right)^i = \frac{1}{4} \times \frac{1}{1 - \frac{1}{4}} = \frac{1}{3}.
\]
Alternatively, we can solve the problem using the law of total probability,
\[
P[\text{home}] = P[\text{home} \mid LR] P[LR] + P[\text{home} \mid LL] P[LL] + P[\text{home} \mid R] P[R].
\]
That is, \( P[\text{home}] = \frac{1}{4} \cdot P[\text{home}] + \frac{1}{4} \). Solving, \( P[\text{home}] = \frac{1}{3} (1 - \frac{1}{4}) = \frac{1}{5} \).

Exercise 17.11.
(a) The analysis is similar to Example 17.5. We want \( P(100, 200, \frac{18}{38}) = (\beta^{200} - \beta^{100})/(\beta^{200} - 1) \), where \( \beta = 0.9 \). The probability to double the money is about 2.66 × 10^{-5}. Essentially 0.

If you bet all your money on red, you double up with probability \( \frac{18}{38} \approx 0.474 \). Much better.

(b) This is an interesting problem and the answer is very counter-intuitive. It sounds like the person furthers from you is likely the last to get the bread, but this is not true. Everyone but yourself is equally likely to be the last. This includes the person sitting next to you as well as the person diametrically opposite. Everyone has a probability \( \frac{1}{k} \) of being the last.

Consider person \( x \). One of \( x \)’s two neighbors gets the bread before the other. In this situation, the bread needs to travel \( n - 2 \) steps to the other neighbor before reaching \( x \) if \( x \) is the last to get the bread (\( n \) is the number of people). That is, we have a random walk with \( k = 1 \) (one step to reach \( x \)) \( n - 2 \) steps in the other direction to reach the other neighbor \( \text{before} \ x \). So, \( k = 1, L = n - 1 \) and \( \beta = 1 \). The probability to reach \( x \) first is \( (L - 1)/L = (n - 2)/(n - 1) \). This is the probability that \( x \) is not last. The probability \( x \) is last is \( 1/(n - 1) \), as claimed.

Important: If you do not believe the analysis, do a Monte Carlo simulation to play out the bread passing routine with 5 people and tabulate how often each person is last.

Chapter 18

Pop Quiz 18.1.
(a) \( X = 2 \) and \( Y = 1 \) are disjoint so \( P[X = 2 \cap Y = 1] = 0 \). \( P[X = 2] \times P[Y = 1] = \frac{3}{8} \times \frac{1}{4} = \frac{3}{32} \). The two do not match. The events are not independent.

(b) \( \{X = 2\} \cap \{Y = 1\} = \{HHHI\} \), hence \( P[X = 2 \cap Y = 1] = \frac{1}{8} \). \( P[X = 2] \times P[Y = 1] = \frac{3}{8} \times \frac{1}{4} = \frac{3}{32} \). The two match. The events are independent.

(c) \( P[X = 2 \mid Y = 0] = \frac{P[X = 2 \cap Y = 0]}{P[Y = 0]} = \frac{\frac{3}{8}}{\frac{5}{8}} = \frac{3}{5} \).

(d) \( P[X \geq 2 \mid Y = 0] = \frac{P[X \geq 2 \cap Y = 0]}{P[Y = 0]} = \frac{\frac{3}{8}}{\frac{5}{8}} = \frac{3}{5} \).

Exercise 18.2. Let us first construct the (non-uniform) probability space and the random variables.

<table>
<thead>
<tr>
<th>outcome</th>
<th>( HHHH )</th>
<th>( HHHT )</th>
<th>( HHTH )</th>
<th>( HHTT )</th>
<th>( HTTH )</th>
<th>( HTTT )</th>
<th>( THHH )</th>
<th>( THHT )</th>
<th>( THTH )</th>
<th>( TTHT )</th>
<th>( TTTH )</th>
<th>( TTTT )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( X_{12} )</td>
<td>16/81</td>
<td>8/81</td>
<td>8/81</td>
<td>8/81</td>
<td>8/81</td>
<td>8/81</td>
<td>1/81</td>
<td>2/81</td>
<td>1/81</td>
<td>0/81</td>
<td>0/81</td>
<td>0/81</td>
</tr>
<tr>
<td>( X_{23} )</td>
<td>2/2</td>
<td>2/2</td>
<td>2/2</td>
<td>2/2</td>
<td>2/2</td>
<td>2/2</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
</tr>
<tr>
<td>( X_{14} )</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
<td>0/2</td>
<td>0/2</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
<td>0/2</td>
<td>0/2</td>
<td>1/2</td>
</tr>
<tr>
<td>( X_{12} + X_{23} )</td>
<td>4/4</td>
<td>4/4</td>
<td>3/3</td>
<td>3/3</td>
<td>2/2</td>
<td>2/2</td>
<td>1/1</td>
<td>1/1</td>
<td>1/1</td>
<td>1/1</td>
<td>1/1</td>
<td>1/1</td>
</tr>
<tr>
<td>( X_{12} + X_{14} )</td>
<td>4/4</td>
<td>4/4</td>
<td>3/3</td>
<td>3/3</td>
<td>2/2</td>
<td>2/2</td>
<td>1/1</td>
<td>1/1</td>
<td>1/1</td>
<td>1/1</td>
<td>1/1</td>
<td>1/1</td>
</tr>
<tr>
<td>( X_{23} + X_{14} )</td>
<td>4/4</td>
<td>4/4</td>
<td>3/3</td>
<td>3/3</td>
<td>2/2</td>
<td>2/2</td>
<td>1/1</td>
<td>1/1</td>
<td>1/1</td>
<td>1/1</td>
<td>1/1</td>
<td>1/1</td>
</tr>
</tbody>
</table>

Using this table, we can compute the probabilities of interest.
(a) (i) \( P[X_{12} \geq 2] = 36/81 = \frac{4}{9} \); (ii) \( P[X_{12} + X_{23} \geq 2] = 66/81 = \frac{22}{27} \). (iii) \( P[X_{12} + X_{14} \geq 2] = 72/81 = \frac{8}{9} \).

(b) (i) \( P[X_{12} \geq 2] = 4/9 \); (ii) \( P[X_{23} \geq 2] = 4/9 \); (iii) \( P[X_{12} \geq 2 \cap X_{23} \geq 2] = 24/81 \neq \frac{4}{9} \times \frac{4}{9} \). Not independent.

(ii) \( P[X_{12} \geq 2] = \frac{4}{9} \); (iii) \( P[X_{34} \geq 2] = \frac{4}{9} \); (iv) \( P[X_{12} \geq 2 \cap X_{34} \geq 2] = 16/81 = \frac{4}{9} \times \frac{4}{9} \). Independent.

Pop Quiz 18.3. The shaded upper-left to lower-right diagonals as shown in the probability space for \( X = 9 \) contain all the outcomes with a particular value of \( X \). The probability is the
number of outcomes in the diagonal divided by 36 (table below). Simplifying the fractions gives the answer.

<table>
<thead>
<tr>
<th>x</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
</tr>
</thead>
</table>

**Pop Quiz 18.4.** The outcome-probabilities in the underlying probability space sum to 1. Every outcome is represented once in the joint probabilities, so the sum of the joint probabilities is 1. Summing the column sums is just another way to sum all the joint probabilities. So the column sums must add to 1. Similarly for the row sums.

**Pop Quiz 18.5.** Yes. The first 5 coin tosses are one experiment; the second 5 tosses are another; and, the 2 dice rolls are a third. X depends only on the first experiment. Y depends only on the 2nd and 3rd experiments which are unrelated to the first experiment.

**Exercise 18.6.**

(a) Start with the probability space and construct the random variables.

<table>
<thead>
<tr>
<th>ω</th>
<th>HHH</th>
<th>HHT</th>
<th>HTH</th>
<th>HTT</th>
<th>THH</th>
<th>THT</th>
<th>TTH</th>
<th>TTT</th>
</tr>
</thead>
</table>

| X(ω) | 3 | 2 | 2 | 1 | 2 | 1 | 1 | 0 |
| Y(ω) | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

| X(ω) + Y(ω) | 4 | 3 | 2 | 1 | 3 | 1 | 1 | 0 |

We use the probability space to derive the joint PDF and the marginals shown on the right. We shaded the event \( X + Y \geq 2 \), from which \( P[X + Y \geq 2] = \frac{7}{27} \). The event \( X \leq 2 \cap X + Y \geq 2 \) has its probabilities in red. The conditional probability is the ratio (red sum)/(shaded sum) = \( \frac{6}{7} \).

(b) In each case, we give the joint PDF and the joint PDF obtained from the product of the marginals. If the two match, the random variables are independent. Otherwise they are not.

(i) \( P_{X_{12}X_{34}} \)

<table>
<thead>
<tr>
<th>X_{12}</th>
<th>P_{X_{12}X_{23}}</th>
<th>X_{12}</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1/9</td>
</tr>
<tr>
<td>1</td>
<td>1/9</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2/9</td>
<td>4</td>
</tr>
</tbody>
</table>

\( P_{X_{12}X_{23}} \neq P_{X_{12}} P_{X_{23}} \), so \( X_{12} \) and \( X_{23} \) are not independent.

(ii) \( P_{X_{12}X_{34}} \)

<table>
<thead>
<tr>
<th>X_{12}</th>
<th>P_{X_{12}X_{34}}</th>
<th>X_{12}</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1/9</td>
</tr>
<tr>
<td>1</td>
<td>1/9</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2/9</td>
<td>4</td>
</tr>
</tbody>
</table>

In all entries, \( P_{X_{12}X_{34}} = P_{X_{12}} P_{X_{34}} \), so \( X_{12} \) and \( X_{34} \) are independent. No surprise because the first 2 coin tosses and the last 2 coin tosses are unrelated experiments.
(iii) \( P_{X \mid 2} X_{4} \) 

<table>
<thead>
<tr>
<th>( X \mid 2 )</th>
<th>1</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( \frac{1}{9} )</td>
<td>( \frac{4}{9} )</td>
</tr>
<tr>
<td>4</td>
<td>( \frac{1}{9} )</td>
<td>( \frac{4}{9} )</td>
</tr>
</tbody>
</table>

No surprise. If two random variables have nothing to do with each other (are independent) then functions of the random variables will also be unrelated.

(c) \( \sum_{x} \sum_{y} P_X(x) P_Y(y) = \sum_{x} P_X(x) \sum_{y} P_Y(y) \). Each individual sum is 1, so the product is 1.

**Exercise 18.7.** We reproduce the outcome tree from Exercise 15.2. Below each outcome in the outcome-tree, is its probability and the value the maximum (the random variable).

We only give the CDF at the values with positive PDF. The PDF can be computed from the CDF by looking at the jumps in the CDF.

(c) Consider \( n \) dice values \( X_1, \ldots, X_n \). In the problem, \( n \) is 10. Note:

\[
\max(X_1, \ldots, X_n) \leq x \Leftrightarrow X_1 \leq x \text{ AND } X_2 \leq x \text{ AND } \cdots \text{ AND } X_n \leq x
\]

The outcomes in the first event are the same as the outcomes in the second. That means,

\[
P[\max(X_1, \ldots, X_n) \leq x] = P[X_1 \leq x \text{ AND } X_2 \leq x \text{ AND } \cdots \text{ AND } X_n \leq x].
\]

Since \( X_1, \ldots, X_n \) are independent, we can compute the RHS by multiplying the individual probabilities. For \( x \in \{1, 2, 3, 4, 5, 6\} \), \( P[X_i \leq x] = \frac{x}{6} \). Therefore,

\[
P[\max(X_1, \ldots, X_n) \leq x] = F_X(x) = \left( \frac{x}{6} \right)^n.
\]

The jumps in \( F_X \) give us the PDF \( P_X \),

\[
P_X(x) = F_X(x) - F_X(x-1) = \frac{x^n - (x-1)^n}{6^n}.
\]

**Pop Quiz 18.8.**

(a) Since I show you the smaller number half the time, you will be wrong half the time.

(b) Using the law of total probability for the two cases you say smaller or larger,

\[
P[\text{you win}] = P[\text{you win | smaller}] P[\text{smaller}] + P[\text{you win | larger}] P[\text{larger}]
\]

\[
= \frac{1}{2} \times \frac{1}{2} + \frac{1}{2} \times \frac{1}{2} = \frac{1}{2}.
\]

(c) I choose (4,5). You always say smaller and win half the time.

(d) Yes! See the discussion following the pop quiz in the text and Exercise 18.9.
Exercise 18.9.

(a) \( P[\text{you win}] = P[E] \cdot P[\text{you win | } E] + P[E^c] \cdot P[\text{you win | } E^c] \) (total probability)

\[ = \frac{1}{2} \cdot 1 + (1 - \frac{1}{2}) \cdot \frac{1}{2} \]  
(if \( E \) occurs you win; if not you win half the time)

\[ = \frac{1}{2} + \frac{1}{2} \cdot P[E] \] (algebra)

\[ = \frac{1}{2} + \frac{1}{2}(H - L) > \frac{1}{2}. \]

(b) Let \( P(X) = p_1, p_2, p_3, p_4 \) with minimum \( p_i < \frac{1}{2} \). I now choose \( L = 1 \) and \( H = i + 1 \). Then \( P[E] = p_i \). And so, \( P[\text{you win}] = \frac{1}{2} + \frac{1}{2} P[E] = \frac{1}{2} + \frac{1}{2} p_i < \frac{1}{2} + \frac{1}{2} \times \frac{1}{2} = \frac{3}{2}. \)

Exercise 18.10.

(a) \( X_i = 1 \) if you get question \( i \) correct; \( P[X_i = 1] = \frac{1}{2} \). The number of correct answers is \( X = X_1 + \cdots + X_{20} \), a Binomial with \( n = 20 \) and \( p = \frac{1}{2} \). We want \( P[X \geq 10] \).

\[ = \sum_{k=10}^{20} \binom{20}{k} \left( \frac{1}{2} \right)^k \left( \frac{1}{2} \right)^{20-k} \approx 0.0026. \]

(b) Let \( X \) be the number of games played between \( A \) and \( B \). The outcomes \( AAAA \) or \( BBBB \) each have probability \( \left( \frac{1}{6} \right)^4 = \frac{1}{1296} \). So \( P[X = 4] = \frac{1}{1296} \).

If the series ends in 5 games, there are two cases: \( A \) wins or \( B \) wins. Each has the same probability.

If \( A \) wins, the series looks like \( xxxxA \), and \( A \) must win 3 games in the first 4.

We have a Binomial with \( n = 4 \) and \( p = \frac{1}{2} \). \( P[k=3] = \left( \frac{1}{5} \right)^3 \frac{1}{2}^1 = \frac{4}{10} \). Therefore \( P[A \text{ wins in } 5] = \frac{4}{10} \times \frac{1}{2} \) (the last \( \frac{1}{2} \) is because \( A \) wins the last game). So,

\[ P[X = 5] = P[A \text{ wins in } 5] + P[B \text{ wins in } 5] = \frac{5}{10}. \]

Similarly, \( A \) wins in 6 games with probability \( \left( \frac{1}{6} \right)^6 \left( \frac{1}{2} \right)^2 \times \frac{1}{2}, \) and

\[ P[X = 6] = P[A \text{ wins in } 6] + P[B \text{ wins in } 6] = \frac{10}{10} = \frac{5}{10}. \]

Lastly, \( A \) wins in 7 games with probability \( \left( \frac{1}{6} \right)^7 \left( \frac{1}{2} \right)^3 \times \frac{1}{2}, \) and

\[ P[X = 7] = P[A \text{ wins in } 7] + P[B \text{ wins in } 7] = \frac{20}{10} = \frac{5}{10}. \]

(c) There are \( \binom{100}{20} \times \binom{80}{30} \times 4^{20} \) sequences with 20 ones and 30 fours (\( \binom{100}{20} \)) ways to choose the ones; of the remaining 80 positions there are \( \binom{30}{20} \) ways to choose the fours; each of the remaining 50 slots can be picked in 4 ways for \( 4^{50} \). The probability of each sequence is \( \left( \frac{1}{6} \right)^{100} \), so

\[ P[20 \text{ ones and 30 fours}] = \left( \frac{100}{20} \right) \times \left( \frac{30}{80} \right) \times 4^{20} \times \left( \frac{1}{8} \right)^{100} \approx 9.226 \times 10^{-6}. \]

Here is another approach similar to the Binomial distribution. We get the multinomial distribution. From counting, there are \( \binom{100}{k_1+k_2+k_3} \) sequences with \( k_1 \) objects of type 1, \( k_2 \) of type 2 and \( k_3 \) of type 3.

That is, there are \( \binom{100}{20,30,50} \) sequences with 20-ones, 30-fours, and 50-other-values (three types of objects). Each one has probability \( \frac{1}{6} \); each four has probability \( \frac{1}{8} \), and each other has probability \( \frac{1}{8} \). Therefore the probability of each such sequence is \( \left( \frac{1}{6} \right)^{20} \cdot \left( \frac{1}{8} \right)^{30} \cdot \left( \frac{1}{8} \right)^{50} \). So, we recover the same probability as:

\[ P[20 \text{ ones and 30 fours}] = \left( \frac{100}{20,30,50} \right) \left( \frac{1}{6} \right)^{20} \cdot \left( \frac{1}{8} \right)^{30} \cdot \left( \frac{1}{8} \right)^{50}. \]

(d) The challenge is to evaluate small numbers, like \((1 - p)^n\). It is numerically more stable to compute the log \( P_X \). So, log \( P_X(0) = n \log(1 - p) \). Having computed log \( P_X(k-1) \), you can use the following update to get log \( P_X(k) \).

\[ \log P_X(k) \leftarrow \log P_X(k-1) + \log \left( \frac{p(n-k+1)}{(1-p)k} \right). \]

Exercise 18.11.

(a) A boy is “success” and the probability of success is \( \frac{1}{2} \). Let \( X \) be the number of trials till success. We want \( P[X \geq 5] \) where \( P_X(t) = \beta(1-p)^t \) with \( p = \frac{1}{2} \) and \( \beta = p/(1-p) = 1 \)

\[ P[X \geq 5] = \sum_{t=5}^{\infty} \left( \frac{1}{2} \right)^t \cdot \frac{1}{1 - 1/2} = \frac{1}{16}. \]
Alternatively, we may observe that $X \geq 5$ if and only if the first 4 children are girls, which happens with probability $(\frac{1}{2})^4$.

(b) You wait $t$ trials for two successes precisely when the $t$th trial is a success and there is one success in trials $1, \ldots, t-1$. There are $t-1$ sequences with one success in the first $t-1$. The probability of each such sequence is $p^2(1-p)^{t-2}$ (two successes and $t-2$ failures), therefore

$$P[X = t] = P_X(t) = (t-1)p^2(1-p)^{t-2} = \beta^2(t-1)(1-p)^t.$$ 

(c) (i) In the first 20 transmissions you have at most 11 successful transmissions. The probability this occurs is Let $X$ be the number of successful transmissions in the first 20 trials; $X \sim B(20, 0.9)$. We want $P[X \leq 11]$,

$$P[X \leq 11] = \sum_{k=0}^{11} \binom{20}{k} (0.9)^k (0.1)^{20-k}.$$ 

(ii) You fail to send any packet with probability $(0.1)^{20}$, so the probability you successfully send at least 1 packet is $1 - (0.1)^{20}$.

### Chapter 19

**Exercise 19.1.** The discussion after the exercise suggests the values you should observe for the Monte Carlo averages. (a) average dice roll is about 3.5. (b) 10 coin tosses yields on average 5 heads. (c) 7.5mm of rain per day on average. (d) The gamblers lose on average $52.63.

**Exercise 19.2.**

(a) $E[X] = \sum_{i=1}^{6} i \times \frac{1}{6} = \frac{1}{6} \times \frac{5}{6} \cdot 6 = \frac{5}{2} = 3 \frac{1}{2}$.

(b) ![Die 1 Value](image)

(c) $E[Y] = \sum_{\omega \in \Omega} P(\omega)Y(\omega) = \sum_{\omega \in \Omega} P(\omega)(aX(\omega) + b) = a \sum_{\omega \in \Omega} P(\omega)X(\omega) + b \sum_{\omega \in \Omega} P(\omega) = aE[X] + b$.

**Pop Quiz 19.3.** We show the probability space with the random variables below.

| Probability Space | $X = |D_1 - D_2|$ |
|-------------------|------------------------|
| ![Die 1 Value](image) | ![Die 2 Value](image) |

To compute the expectations, we weight each random-variable value by the corresponding probability and add. For $D_1 - D_2$, every positive value has a corresponding negative value which cancels, so $E[D_1 - D_2] = 0$. The cancelation does not occur for $E[|D_1 - D_2|]$,

$$E[|D_1 - D_2|] = \frac{1}{36} \times 2 \times (5 \times 1 + 4 \times 2 + 3 \times 3 + 2 \times 4 + 1 \times 5) = \frac{70}{36} = 1 \frac{17}{18}.$$ 

**Exercise 19.4.**

502
(a) \( n = 20 \) and \( p = \frac{1}{2} \), so \( \mathbb{E}[X] = np = 10 \). The expected number of heads is 10.
(b) \( n = 20 \) and \( p = \frac{5}{6} \), so \( \mathbb{E}[X] = np = 4 \). The expected number of correct answers is 4.
(c) \( \mathbb{E}[X(X-1)] = \sum_{k=0}^{n} k(k-1) \binom{n}{k} p^k(1-p)^{n-k} = \sum_{k=2}^{n} k(k-1) \binom{n}{k} p^k(1-p)^{n-k} \). Observe that
\[
k(k-1) \binom{n}{k} = \frac{k(n-1)!}{k!(n-k)!} = \frac{n!}{(k-2)!(n-k)!} = n(n-1)(n-2)! = n(n-1) \binom{n-2}{k-2}.
\]
Using this identity in the expression for \( \mathbb{E}[X(X-1)] \),
\[
\mathbb{E}[X(X-1)] = \sum_{k=2}^{n} n(n-1) \binom{n-2}{k-2} p^k(1-p)^{n-k} - \binom{n}{k} p^k(1-p)^{n-k} = n(n-1)p^2 \sum_{k=2}^{n} \binom{n-2}{k-2} p^{k-2}(1-p)^{n-k} - \binom{n}{k} p^k(1-p)^{n-k} = n(n-1)p^2 (p+1-p)^{n-2} = n(n-1)p^2 (p+1-p) = n(n-1)p^2 (p+1-p).
\]

**Exercise 19.5.**
(a) Hitting the bulls-eye is success. The bulls-eye area is \( \frac{1}{100} \) of the area of the board, so \( p = \frac{1}{100} \).
Therefore, the expected number of darts you throw is \( \frac{1}{p} = 100 \).
(b) A 5-pack contains no EX with probability \( 0.99^5 \approx 0.95 \). So a 5-pack contains an EX (success) with probability \( p \approx 0.05 \). You expect to buy \( \frac{1}{p} \approx 20 \) 5-packs to get an EX.
(c) If you pay \( x \), with probability \( 10^{-7} \) you win \( 10^6 \) and with probability \( 1 - 10^{-7} \) you lose \( x \), so your expected profit is \( 10^6 \times 10^{-7} - x(1 - 10^{-7}) \). You would only play if you made a profit so \( x < \frac{10^6}{10^7} \). You would pay approximately 10¢ to play.
(d) \( i \) We may assume the first child is a boy (the argument is identical if the first is a girl).
You are now waiting for a girl with success probability \( p = \frac{1}{2} \). Therefore you expect 2 trials to get a girl, for a total of 3.
\( ii \) You expect to wait 2 trials for the first boy and 2 more for the second for a total of 4 kids. Let’s compute this from the PDF computed in Exercise 18.11 for the waiting time to two successes with success probability \( p \).
\[
\mathbb{E}[X] = \sum_{t=1}^{\infty} t(t-1)\left(\frac{1}{2}\right)^t.
\]
We need to compute this infinite sum. Note that the first term in the sum is zero and
\[
\frac{1}{2} \mathbb{E}[X] = \sum_{t=1}^{\infty} t(t-1)\left(\frac{1}{2}\right)^{t+1} = \sum_{t=2}^{\infty} (t-1)(t-2)\left(\frac{1}{2}\right)^t.
\]
Subtracting (***) from (***), the LHS is \( \mathbb{E}[X] - \frac{1}{2} \mathbb{E}[X] = \frac{1}{2} \mathbb{E}[X] \) and we have:
\[
\frac{1}{2} \mathbb{E}[X] = \sum_{t=2}^{\infty} (t-1)(t-2)\left(\frac{1}{2}\right)^t = \sum_{t=2}^{\infty} 2(t-1)\left(\frac{1}{2}\right)^t = \sum_{t=2}^{\infty} (t-1)\left(\frac{1}{2}\right)^{t-1}.
\]
The last sum is \( \sum_{t=1}^{\infty} t\left(\frac{1}{2}\right)^t \) which equals 2 by Lemma 19.6 on page 287, so we get \( \mathbb{E}[X] = 4 \).

**Exercise 19.6.**
(a) Conditioned on \( D_1 + D_2 \geq 4 \), there are now 33 outcomes, each having probability \( \frac{1}{33} \). So, \( \mathbb{E}[D_1 + D_2 | D_1 + D_2 \geq 4] = \frac{2 \times 14 \times 4 \times 5 \times 5 \times 6 \times 7 \times 7 \times 5 \times 8 \times 4 \times 9 \times 3 \times 10 \times 2 \times 11 \times 1 \times 12}{33} = \frac{244}{33} \approx 7.4 \).
(b) \( i \) The relevant outcomes are \( \{0, 2, 4, \ldots, 20\} \).
\[
\mathbb{P}[X = 2i \text{ even}] = \frac{\mathbb{P}[X = 2i \text{ even}]}{\mathbb{P}[\text{even}]} = \frac{1}{\mathbb{P}[\text{even}]} \binom{20}{2i} \left(\frac{1}{2}\right)^{2i} = \frac{1}{\mathbb{P}[\text{even}]} \binom{20}{2i} \left(\frac{1}{2}\right)^{2i}.
\]
We can compute \( \mathbb{P}[\text{even}] = \left(\frac{1}{2}\right)^{20} \sum_{i=0}^{20} \binom{20}{i} = \left(\frac{1}{2}\right)^{20} \approx 2^{10} = \frac{1}{2} \).
We can compute \( \mathbb{P}[\text{even}] = \left(\frac{1}{2}\right)^{20} \sum_{i=0}^{20} \binom{20}{i} = \left(\frac{1}{2}\right)^{20} \times 2^{10} = \frac{1}{2} \),
where we used \( \sum_{i=0}^{20} \binom{20}{i} = 2^{20} \) (the same is also true for the sum over the odd Binomial
We want \( x \geq 25 \). We need the conditional probability 
\[
\Pr[X \geq 25 | \beta(k + 1)] = \frac{\Pr[X \geq 25 \cap X \geq k + 1]}{\Pr[X \geq k + 1]} = \frac{\beta(1 - p)^k}{(1 - p)^k}.
\]

Because \( \Pr[X \geq k + 1] \) is the probability to fail on the first \( k \) trials which is \((1 - p)^k\). Therefore,
\[
\Pr[X \geq k + 1] = \frac{\beta}{(1 - p)^k} \sum_{t=k+1}^{\infty} t(1 - p)^t = \frac{\beta}{(1 - p)^k} \sum_{t=1}^{\infty} (t + k)(1 - p)^{t+k} = \frac{1}{p} + k.
\]

(e) \( \Pr[X \geq 25] = \sum_{x \geq 25} \frac{xP_X(x)}{\Pr[X \geq 25]} \) and \( \Pr[X \geq 17] = \sum_{x \geq 17} \frac{xP_X(x)}{\Pr[X \geq 17]} \). So,
\[
\Pr[X \geq 25] - \Pr[X \geq 17] = \left( \frac{\Pr[X \geq 17]}{\Pr[X \geq 25]} \sum_{x \geq 25} xP_X(x) - \Pr[X \geq 25] \sum_{x \geq 17} xP_X(x) \right) \frac{\Pr[X \geq 17]}{\Pr[X \geq 25]} \frac{\Pr[X \geq 25]}{\Pr[X \geq 17]}.
\]

We show that the numerator is positive: Let \( P_1 = \Pr[17 \leq X < 25] \); \( P_2 = \Pr[X \geq 25] \); \( S_1 = \sum_{x=17}^{24} xP_X(x) \); and, \( S_2 = \sum_{x=25}^{\infty} xP_X(x) \). Note: \( S_2 > 25P_2 \) and \( S_1 < 24P_1 \). We have,
\[
\Pr[X \geq 17] \sum_{x \geq 25} xP_X(x) - \Pr[X \geq 25] \sum_{x \geq 17} xP_X(x)
= (P_1 + P_2)S_2 - P_2(S_1 + S_2)
= P_1S_2 - P_2S_1
> 25P_1P_2 - 24P_2P_1
> 0.
\]
Pop Quiz 19.7.
(a) Definition of expected value.
(b) By the law of total probability, \( P(X = x) = P(A) P(X = x | A) + P(A^c) P(X = x | A^c) \).
(c) \( P(A) \) and \( P(A^c) \) are independent of \( x \) and can be pulled out of the sum (constant rule).
(d) Definition of the conditional expectation.

Exercise 19.8.
(a) The logic is similar to the case of 3 dice. Let \( X_1 \) be the first die, and \( X_3 \) the sum of the remaining 3 dice. The dice are independent, so \( P[X_3 = x_3 | X_1 = x_1] = P_{X_3}(x_3) \). By the law of total probability,

\[
E[X] = \sum_{i=1}^{6} E[X_1 | X_1 = i] P[X_1 = i].
\]

Since \( E[X | X_1 = i] = \sum_{x_3}(i + x_3)P_{X_3}(x_3) = i + E[X_3] = i + 10.5 \), we have

\[
\frac{1}{6} E[X] = \sum_{i=1}^{6}(i + 10.5) = \frac{1}{6} \sum_{i=1}^{6} \sum_{x_3}(i + x_3)P_{X_3}(x_3) = \frac{1}{6} \sum_{i=1}^{6} gi + \frac{1}{6} E[X] = \sum_{i=1}^{6} 10.5 = 3.5 + 10.5 = 14.
\]

The expected sum of 4 dice is 4 times the expected value of one die.
(b) \( E[X] = E[X | B] P[B] + E[X | G] P[G] \). Let \( p \) be the probability of a boy. If you start with a boy, you are now waiting for a girl, so \( E[X | B] = 1 + 1/(1 - p) \). If you start with a girl, you are now waiting for a boy, so \( E[X | G] = 1 + 1/p \). So,

\[
E[X] = \left( 1 + \frac{1}{1 - p} \right) p + \left( 1 + \frac{1}{p} \right) (1 - p) = 1 + \frac{p}{1-p} + \frac{1-p}{p}.
\]

When \( p = \frac{1}{2} \), you expect to 3 kids till you get a boy and a girl.
(c) \( E[X] = E[X | H] P[H] + E[X | T] P[T] = 7p + 10.5(1 - p) = 10.5 - 3.5p \), where \( p = P[H] \).
(d) \( E[X] = E[X | \text{fair}] P[\text{fair}] + E[X | \text{2-heads}] P[\text{2-heads}] + E[X | \text{2-heads}] P[\text{2-heads}] = 5 \times \frac{9}{10} + 10 + \frac{10}{10} = 5 \frac{1}{2} \).
(e) \( E[X^2] = E[X^2 | \text{success}] P[\text{success}] + E[X^2 | \text{fail}] P[\text{fail}] \)

Exercise 19.9.
(a) For \( n > 1 \), let \( A_i \) be the event that the pivot is picked as the \( i \)th smallest number, \( i = 1, \ldots, n \).

\( P[A_i] = \frac{1}{n} \). By the law of total probability,

\[
T_n = E[\text{runtime}(n)] = \sum_{i=1}^{n} E[\text{runtime} | A_i] P[A_i] = \frac{1}{n} \sum_{i=1}^{n} E[\text{runtime} | A_i].
\]

Given \( A_i \), the runtime is \( n + 1 \) plus the runtime on the left list of size \( i - 1 \) plus the runtime on the right list of size \( n - i \).

\[
\text{runtime given } A_i = n + 1 + \text{runtime}(i - 1) + \text{runtime}(n - i).
\]

Taking the expectation, \( E[\text{runtime} | A_i] = (n + 1) + T_{i-1} + T_{n-i} \). Therefore,

\[
T_n = \frac{1}{n} \sum_{i=1}^{n} ((n+1) + T_{i-1} + T_{n-i}) = n + 1 + \frac{1}{n} (T_0 + T_{n-1} + T_1 + T_{n-2} + \cdots + T_{n-2} + T_1 + T_{n-1} + T_0).
\]

The sum contains two copies of each \( T_i \). Since \( T_0 = 0 \), we have

\[
T_n = n + 1 + \frac{2}{n} \sum_{i=1}^{n-1} T_i.
\]

(b) Using (a), \( T_2 = 3 + \frac{2}{2} T_1 = 4 \). Rewriting the recursion in (a),

\[
nT_n = n(n + 1) + 2 \sum_{i=1}^{n-1} T_i.
\]

Similarly, for \( n > 2 \), we have

\[
(n-1)T_{n-1} = (n-1)n + 2 \sum_{i=1}^{n-2} T_i.
\]

Subtracting this equation from the previous,

\[
nT_n - (n - 1)T_{n-1} = n(n + 1) - (n - 1)n + 2 \sum_{i=1}^{n-1} T_i - 2 \sum_{i=1}^{n-2} T_i = 2n + 2T_{n-1}.
\]

Rearranging gives \( nT_n = (n + 1)T_{n-1} + 2n \). Dividing both sides by \( n \) gives the desired result.
(c) Deriving the upper bound is interesting. Given the bound, proving it by induction is good practice. We need two facts: $1 + x \leq e^x$; $e^{H_n} \geq n + 1$ (which we proved using the integration method on page 119.) One can verify the bound for $T_1$ and $T_2$. Suppose the bound holds for $T_n$ and apply this induction hypothesis to $T_{n+1}$:

$$T_{n+1} = (1 + \frac{1}{n+1})T_n + 2 \leq 2(1 + \frac{1}{n+1})e^{H_n} + 2 \leq 2e^{1/(n+1)}e^{H_n} + 2 = 2H_ne^{H_{n+1}} + 2.$$

Writing $H_n = H_{n+1} - \frac{1}{n+1}$, we get that

$$T_{n+1} \leq 2(H_{n+1} - \frac{1}{n+1})e^{H_{n+1}} + 2 = 2H_{n+1}e^{H_{n+1}} + 1 - \frac{1}{n+1}.$$  

e^{H_{n+1}} > n + 2 implies $\frac{1}{n+1}e^{H_{n+1}} > 1$, or $1 - \frac{1}{n+1}e^{H_{n+1}} < 0$. Therefore,

$$T_{n+1} \leq 2H_{n+1}e^{H_{n+1}} + 1 - \frac{1}{n+1}e^{H_{n+1}} \leq 2H_{n+1}e^{H_{n+1}}.$$

How did we derive the upper bound? We unfolded the recursion:

$$T_n = (1 + \frac{1}{n})T_{n-1} + 2$$

$$(1 + \frac{1}{n})(1 + \frac{1}{n-1})T_{n-2} + 2(1 + \frac{1}{n})$$

$$(1 + \frac{1}{n})(1 + \frac{1}{n-1})(1 + \frac{1}{n-2})T_{n-3} + 2(1 + \frac{1}{n})(1 + \frac{1}{n-1})$$

$$\vdots$$

$$(1 + \frac{1}{n})\cdot \cdots (1 + \frac{1}{2})T_3 = (1 + \frac{1}{n})(1 + \frac{1}{n-1})\cdots (1 + \frac{1}{2})T_3 + 2(1 + \frac{1}{n})\cdots (1 + \frac{1}{2}).$$

Now equate the sum of the left hand sides to the sum of the right hand sides,

$$T_n = (1 + \frac{1}{n})\cdots (1 + \frac{1}{2})T_2 + 2(1 + \frac{1}{n})\cdots (1 + \frac{1}{2})T_3 + (1 + \frac{1}{n})\cdots (1 + \frac{1}{2}).$$

The first term is bounded by $e^{1/n}e^{1/(n-1)}\cdots e^{1/2} = e^{H_n-H_2}$. 

The second term is a sum of terms of the form $a_k$, where $a_k = (1 + \frac{1}{n})\cdots (1 + \frac{1}{2})$. Using $\ln(1+x) \leq x$ (because $1+x \leq e^x$),

$$\log a_k \leq \sum_{i=k}^{n} \frac{1}{i} = H_n - H_{k-1}.$$  

which implies

$$T_n \leq T_2e^{H_n-H_2} + 2\sum_{k=3}^{n} e^{H_n-H_k} = T_2e^{H_n-H_2} + 2e^{H_n}\sum_{k=3}^{n} e^{-H_k}.$$  

Since $H_k \geq \ln{k}$, $e^{-H_k} \leq \frac{1}{k}$ and so $\sum_{k=3}^{n} e^{-H_k} \leq \sum_{k=3}^{n} \frac{1}{k} = H_n - H_2$. We conclude that

$$T_n \leq T_2e^{H_n-H_2} + 2e^{H_n}(H_n - H_2) = 2He^{H_n} + e^{H_n}(T_2e^{H_2} - 2H_2).$$

Since $T_2 = 4$, you may verify that $T_2e^{H_2} - 2H_2 < 0$ which gives the bound.

Chapter 20

Exercise 20.1.

(a) $X$ is the sum of the dice, $X = X_1 + \cdots + X_n$. $X_i$ is an $r_i$-sided dice, so $E[X_i] = \frac{1}{r_i}(1 + \cdots + r_i) = \frac{1}{r_i} \times \frac{1}{r_i}r_i(r_i + 1) = \frac{1}{r_i}(r_i + 1).$ By linearity of expectation,

$$E[X] = \sum_{i=1}^{n} E[X_i] = \frac{1}{2} \sum_{i=1}^{n} (1 + r_i) = \frac{n}{2} + \frac{1}{2} \sum_{i=1}^{n} r_i.$$

(b) Let $X_i$ indicate (0 or 1) whether trial $i$ is a success. $X_i$ is a Bernoulli indicator random variable with success probability $p_i$ and $E[X_i] = p_i$. The number of successes $X = X_1 + \cdots + X_n$. By linearity of expectation, $E[X] = \sum_{i=1}^{n} E[X_i] = \sum_{i=1}^{n} p_i.$

(c) $X_i$ is the number of trials from the $(i-1)$th success to the $i$th and $X = X_1 + \cdots + X_n$. $E[X_i] = 1/p_i$. By linearity of expectation, $E[X] = \sum_{i=1}^{n} 1/p_i.$

(d) Computing the PDF is very challenging, let alone computing the expectation from the PDF.

Pop Quiz 20.2. We need to count the possible outcomes. There are 6 choices for the first die. If die 1 is $i$, there are $i$ further rolls, so there are $6^i$ possible outcomes if the first die is $i$. Therefore, the number of possible outcomes is $6^1 + \cdots + 6^6 = 55986$.

The probability space is not uniform. The outcome $(1, 1)$ has probability $1/6^2$, but the outcome $(2, 1, 1)$ has probability $1/6^3$.

You could create a uniform outcome space by tossing 7 dice. You would then define the equivalent random variable $X_1$ as the first roll and $X_2$ as the sum of the next $X_1$ rolls.

Exercise 20.3.
(a) For cases $X = 1, 2, \ldots, 100$, total probability gives

$$P_Y(y) = \sum_{x=1}^{100} P_Y(y \mid x) P_X(x) = \frac{1}{100} \sum_{x=1}^{100} \frac{1}{y}$$

because: $P_X(x) = 1/100 (X \sim U[100])$; $P_Y(y \mid x) = 0$ for $x < y$; $P_Y(y \mid x) = 1/x$ for $x \geq y$. ($Y \sim U[x]$). The sum is $H_{100}$ for $y = 1$ and $H_{100} - H_{y-1}$ for $y > 1$, where $H_n$ is the $n$th harmonic number. So,

$$P_Y(y) = \begin{cases} \frac{H_{100}}{100} & y = 1; \\ \frac{(H_{100} - H_{y-1})}{100} & y \in \{2, \ldots, 100\} \end{cases}$$

If you wish to verify that the probabilities sum to 1, you may do so, but you will need the Harmonic sum $\sum_{i=1}^{n} H_n = (n+1)H_n - n$ (you can prove this by induction). Computing the expected value from this PDF is a torturous exercise in summation, but since we are young, let’s go for it. In the process we show how to compute Harmonic sums, which occur frequently in computer science. We will do the computation for general $n$ (in our case $n = 100$).

$$E[Y] = \sum_{i=1}^{n} i P_Y(i) = \frac{1}{n} \left( H_n + \sum_{i=2}^{n} i (H_n - H_{i-1}) \right) = \frac{1}{n} \left( H_n \sum_{i=1}^{n} i - \sum_{i=2}^{n} i H_{i-1} \right).$$

The first sum is $\frac{1}{2} n(n+1)$. The second sum is an example of a Harmonic sum.

$$\sum_{i=2}^{n} i H_{i-1} = 2 \times H_1 + 3 \times H_2 + 4 \times H_3 + \cdots + n \times H_{n-1} = \frac{n(n+1)}{2} + \frac{n}{3} + \frac{n}{4} + \cdots + \frac{n}{n-1}$$

Instead of summing rows, let’s sum columns (shaded). The first column is $(2 + \cdots + n)/1 = \frac{1}{2} (n(n + 1) - 1(1 + 1))/1$; the second column is $(3 + \cdots + n)/2 = \frac{1}{2} (n(n + 1) - 2(2 + 1))/2$; the $i$th column is $((i+1) + \cdots + n)/i = \frac{1}{2} (n(n + 1) - i(i + 1))/i$. Therefore,

$$\sum_{i=2}^{n} i H_{i-1} = \frac{1}{2} \sum_{i=1}^{n-1} \frac{n(n+1) - i(i+1)}{i}$$

$$= \frac{n(n+1)}{2} H_{n-1} - \frac{1}{2} \sum_{i=1}^{n-1} (i + 1)$$

$$= \frac{n(n+1)}{2} H_{n-1} - \frac{1}{2} \sum_{i=1}^{n-1} (i + 1) = \frac{n(n+1)}{2} H_{n-1} - \frac{1}{2} \left( \frac{1}{2} n(n+1) - 1 \right).$$

(You may use this technique to compute the Harmonic sum $\sum_{i=1}^{n} H_n$.) For $E[Y]$ we get

$$E[Y] = \frac{1}{n} \left( \frac{1}{2} n(n+1) H_n - \frac{1}{2} n(n+1) H_{n-1} + \frac{1}{2} n(n+1) - \frac{1}{2} \right)$$

$$= \frac{1}{n} \left( \frac{1}{2} n(n+1) \underbrace{(H_n - H_{n-1}) + \frac{1}{2} n(n+1) - \frac{1}{2}}_{1/n} \right) = \frac{n + 3}{4}.$$

In our case, $n = 100$, so $E[Y] = 25 \frac{3}{4}$.

(b) $Y \sim U[X]$. By Theorem 19.5 on page 287, $E[Y \mid X] = \frac{1}{2} (X + 1)$. By iterated expectation,

$$E[Y] = E_X(E[Y \mid X]) = E[\frac{1}{2} (X + 1)] = \frac{1}{2} + \frac{1}{4} E[X] = 25 \frac{3}{4},$$

where the last step follows because $X \sim U[100]$, so $E[X] = \frac{1}{2} (100 + 1)$. How much easier!

Exercise 20.4.
(a) \( Y \) depends on \( Z \). If \( Z = 0 \), \( Y = 0 \), so \( E[Y | Z = 0] = 0 \). If \( Z = 1 \), \( Y \) has the same PDF as \( X \), so \( E[Y | Z = 1] = E[X] \) and \( E[Y^2 | Z = 1] = E[X^2] \). We can summarize both cases as
\[
E[Y | Z] = E[X] \cdot Z;
\]
\[
E[Y^2 | Z] = E[X^2] \cdot Z.
\]
(b) \( E[Y] = E[Z | E[Y | Z]] = E[Z | E[X] \cdot Z] = E[X] \cdot E[Z] = (1 - p) E[X] \)
(\text{Remember that } E[X] \text{ is just a number, so it can be pulled outside the expectation w.r.t. } Z.)

**Why bother to derive something we already derived using total expectation?**

- It always helps to see an old result derived using a new tool.
- This derivation using iterated expectation is easier to apply to more complicated situations.

**Pop Quiz 20.5.**

- (a) \( X \) is fixed with respect to the inner expectation that is w.r.t. \( Y \).
- (b) \( Y \) is independent of \( X \), so the PDF for \( Y \) given \( X \) is unchanged, so the conditional expectation is the same as the unconditional expectation.
- (c) \( E[Y] \) is a number independent of \( X \) that can be pulled outside the expectation.

**Exercise 20.6.**

- (a) The information \( X_1 + X_2 = 9 \) introduces a dependency between \( X_1 \) and \( X_2 \). The possible outcomes are \((3, 6), (4, 5), (5, 4), (6, 3)\), each with conditional probability \( \frac{1}{4} \). The conditional expectation of the product is \( \frac{1}{4} (18 + 20 + 20 + 18) = 19 \). The conditional expectation of each roll is \( \frac{1}{4} (3 + 4 + 5 + 6) = 4.5 \) and \( 4.5^2 = 20.25 \), so the conditional expectation of the product does not equal the product of conditional expectations, even though the random variables started out independent.
- (i) \( E[X_1/X_2] = \frac{1}{4} \left( \frac{1}{4} + \frac{1}{4} + \frac{1}{4} + \frac{1}{4} \right) \approx 0.41 \), where as \( 1/E[X_1] = 1/3.5 \approx 0.29 \). Not equal.
- (ii) By the law of total expectation, \( E[X_1/X_2] = \sum_{i=1}^{6} E[X_1/X_2 | X_1 = i] \cdot P[X_1 = i] \). Since

\[
E[X_1/X_2 | X_1 = i] = \frac{1}{i} \cdot E[1/X_2],
\]

it follows that
\[
E[X_1/X_2] = \frac{1}{6} \sum_{i=1}^{6} i \cdot E[X_1] = \frac{1}{1/X_2} \approx 1.43.
\]
- (iii) In (ii) we showed that \( E[X_1/X_2] E[X_1] = E[1/X_2] \).

**Exercise 20.7.** The main insight to solving problems like this is to realize that you can "reparameterize" a sum (just like changing variables in a double integral) to change the order in which the terms are added. Here, we let \( n \) be the sum of \( k \) and \( i \), so \( n = k + i \). The possible values of \( n \) are \( 0, 1, \ldots, r \). Given \( n \), the possible values of \( k \) are \( 0, 1, \ldots, n \); given \( n, k, i = n - k \). Therefore,
\[
\sum_{k=0}^{r} \sum_{i=0}^{r-k} f(k, i) = \sum_{k=0}^{n} \sum_{i=0}^{n-k} f(k, n-k).
\]
The identity above holds for any function \( f(k, i) \); every term on the left is accounted for on the right and vice-versa. Using this identity,
\[
\sum_{k=0}^{r} \sum_{i=0}^{r-k} (-1)^{i} = \sum_{n=0}^{r} \sum_{k=0}^{n} \frac{(-1)^{n-k}}{k!(n-k)!} = \sum_{n=0}^{r} \frac{(-1)^{n}}{n!} \sum_{k=0}^{n} \frac{n!}{k!(n-k)!} (-1)^{k} = \sum_{n=0}^{r} (-1)^{n} \sum_{k=0}^{n} \binom{n}{k} (-1)^{k}.
\]
By the Binomial theorem, \( \sum_{k=0}^{n} \binom{n}{k} (-1)^{k} = (-1 + 1)^{n} = 0 \) when \( n > 0 \). So, all the terms are 0 except for \( n = 0 \), which is \( (-1)^{0}/0! = 1 \).

**Pop Quiz 20.8.** \( y_1 \) is the minimum possible temperature, so it cannot possibly break the record.

**Pop Quiz 20.9.** \( 1 + \frac{1}{2} \sum_{i=1}^{n} \frac{n}{n-1} = 1 + \frac{1}{2} \sum_{i=1}^{n} \left( \frac{n}{n-1} - 1 \right) = 1 - \frac{n+1}{n} + \frac{1}{n} = \frac{1}{n} + \frac{1}{n-1} + \cdots + \frac{1}{1} = H_n. \)

**Exercise 20.10.**

- (a) Let \( X_i \) be an indicator that \( i \) is picked. \( P[X_i = 0] = (1 - 1/n)^m \) (by independence, because \( i \) is not picked with probability \( 1 - 1/n \)). \( X_i \) is Bernoulli, so \( E[X_i] = 1 - (1 - 1/n)^m \). The number of distinct elements is \( X = X_1 + \cdots + X_n \). By linearity of expectation,
\[
E[X] = \sum_{i=1}^{n} E[X_i] = n \left( 1 - \left( 1 - \frac{1}{n} \right)^m \right) = n \left( n - 1 \right)^m.
\]
(b) Let $X_i$ indicate whether white ball $i$ is picked, $i = 1, \ldots, a$. There are $\binom{a+b}{k}$ ways to choose $k$ balls (without replacement), each equally likely. If ball $i$ is not picked, there are $\binom{a+b-1}{k-1}$ ways. So, $P[X_i = 0] = \binom{a+b-1}{k-1}/\binom{a+b}{k} = 1 - k/(a+b)$ and $E[X_i] = k/(a+b)$. The number of white balls picked is $X = X_1 + \cdots + X_a$. By linearity of expectation,
\[ E[X] = \sum_{i=1}^{a} E[X_i] = \frac{ak}{a+b}. \]

(c) Obtaining the expectations from the PDF is indeed a challenge for two reasons. First we must get the PDF. This is going require all our expertise in counting. Then, we must compute the expectation from the PDF which will require heavy duty summation. Why go through the effort? Because the expectation is not the only reason to get the PDF. What if you wanted to know how large an $m$ you should use if you want to see at least half the objects in part (a)?

(a) We need $P(k) = \text{probability that } k \text{ distinct elements are sampled.}$ There are $n^m$ $m$-sequences of the $n$ objects. How many of these contain exactly $k$ distinct objects?

First choose the $k$ elements to sample in $\binom{n}{k}$ ways. Now from these elements, construct your $m$-sequence, with the condition that each of the $k$ elements are used at least once. The number of $m$-sequences of the $k$ objects is $k^m$. Let $A_i$ be the sequences that do not use element $i$. The number of sequences using all $k$ elements is $k^m - |A_1 \cup A_2 \cup \cdots \cup A_k|$. Since $|\ell\text{-way intersection of } A_i| = (k - \ell)^m$, by inclusion-exclusion,
\[ |A_1 \cup A_2 \cup \cdots \cup A_k| = \sum_{\ell=1}^{k} (-1)^{\ell+1} \binom{k}{\ell} (k - \ell)^m. \]

There are $k^m - \sum_{\ell=0}^{k} (-1)^{\ell+1} \binom{k}{\ell} (k - \ell)^m = \sum_{\ell=0}^{k} (-1)^{\ell} \binom{k}{\ell} (k - \ell)^m$ sequences that use all $k$ objects. Dividing by $n^m$ gives
\[ P(k) = \frac{n^m}{k} \sum_{\ell=0}^{k} (-1)^{\ell} \binom{k}{\ell} (k - \ell)^m = \binom{n}{k} \sum_{\ell=0}^{k} (-1)^{k-\ell} \binom{k}{\ell} \left(\frac{\ell}{n}\right)^m, \]

where $k = 1, 2, \ldots, n$. The second expression changes summation index to $k - \ell$ and uses $\binom{k}{k-\ell} = \binom{k}{\ell}$. Note that if $k > m$, then $P(k)$ should be 0 and this is indeed the case.

**Lemma 30.7.** If $k > m$, then $\sum_{\ell=0}^{k} (-1)^{\ell} \binom{k}{\ell} \ell^m = 0$.

**Proof.** We use strong induction on $m$. When $m = 0$, $\sum_{\ell=0}^{k} (-1)^{\ell} \binom{k}{\ell} = (1 - 1)^k = 0$. Assume for $m \geq 0$ that $\sum_{\ell=0}^{k} (-1)^{\ell} \binom{k}{\ell} \ell^x = 0$ for all $x \leq m$ and $k > x$. Consider $\sum_{\ell=0}^{k} (-1)^{\ell} \binom{k}{\ell} \ell^{m+1}$ with $k > m + 1$. Since $m \geq 0$, the first term is 0, so we have
\[ \sum_{\ell=1}^{k} (-1)^{\ell} \binom{k}{\ell} \ell^{m+1} = \sum_{\ell=1}^{k} (-1)^{\ell} \binom{k}{\ell} \ell^{m} = \sum_{\ell=1}^{k} (-1)^{\ell} \binom{k}{\ell} (k - \ell)^m \cdot \ell. \]

(We used $\ell \binom{k}{\ell} = k \binom{k-1}{\ell-1}$.) Summing from 0 to $k - 1$ instead of 1 to $k$, we get
\[ -k \sum_{\ell=0}^{k-1} (-1)^{\ell} \binom{k-1}{\ell} (\ell + 1)^m = -k \sum_{\ell=0}^{k-1} (-1)^{\ell} \binom{k-1}{\ell} \sum_{\ell=0}^{m} \binom{m}{\ell} \ell^{\ell}. \]

(We used the Binomial theorem for $(\ell + 1)^m$ and in the last step we changed the order of summing.) Since $k > m + 1$, $k - 1 > m$ and the right sum is zero for every $i \in \{0, \ldots, m\}$ by the induction hypothesis, concluding the proof.

The use of this lemma is that we may compute $E[X]$ as $\sum_{k=1}^{a} kP(k)$ without regard to $m$ because if $k > m$ then $P(k) = 0$.

**Exercise:** We know that $\sum_{k=1}^{a} P(k) = 1$ since it is a PDF. Prove it.
To work the exercise, you will need to use techniques as we are going to display. We want
\[
\sum_{k=1}^{n} k P(k) = \sum_{k=1}^{n} (k - n + n) P(k)
\]
\[
= n - \sum_{k=1}^{n} (n - k) P(k) \quad \text{use: } \sum_{k=1}^{n} P(k) = 1
\]
\[
= n - \sum_{k=1}^{n} \sum_{\ell=1}^{k} (-1)^{k-\ell} (n - k) \binom{n}{\ell} \left(\frac{k}{n}\right)^\ell m
\]
\[
= n - \sum_{\ell=1}^{n} \sum_{k=\ell}^{n} (-1)^{k-\ell} (n - k) \binom{n}{\ell} \left(\frac{n-\ell}{n-k}\right)^\ell m
\]
\[
= n - \sum_{\ell=1}^{n} \sum_{k=\ell}^{n} (-1)^{k-\ell} (n - \ell) \binom{n}{\ell} \left(\frac{n-\ell}{n-k}\right)^\ell m
\]
In the last step we used \((n - k) \binom{n-\ell}{n-k-1} = (n - \ell) \binom{n-\ell-1}{n-k-1}\) and dropped the term with \(k = n\) because it is zero. Now move all terms involving only \(\ell\) outside the inner sum,
\[
\sum_{k=1}^{n} k P(k) = n - \sum_{\ell=1}^{n-1} (-1)^{-\ell} (n - \ell) \binom{n}{\ell} \left(\frac{n-\ell}{n-k}\right)^\ell m \sum_{k=\ell}^{n} (-1)^{k-\ell} \binom{n-\ell-1}{n-k-1}
\]
\[
= n - \sum_{\ell=1}^{n-1} (-1)^{-\ell} (n - \ell) \binom{n}{\ell} \left(\frac{n-\ell}{n-k}\right)^\ell m \sum_{i=0}^{n-\ell-1} (-1)^{n-1-i} \binom{n-\ell-1}{n-1-i}
\]
\[
= n - \sum_{\ell=1}^{n} (-1)^{n-1-\ell} (n - \ell) \binom{n}{\ell} \left(\frac{n-\ell}{n-k}\right)^\ell m \sum_{i=0}^{n-\ell-1} (-1)^{n-1-i} \binom{n-\ell-1}{n-1-i}
\]
The inner sum is an alternating sum of Binomial coefficients which we know is 0, unless \(n - 1 - \ell = 0\) in which case the inner sum is 1. Therefore, the entire sum for the expectation collapses to one term, the one with \(\ell = n - 1\):
\[
\sum_{k=1}^{n} k P(k) = n - (-1)^{n-1-n} (n - 1) \binom{n}{n-1} \left(\frac{n-1}{n}\right)^m = n - (1 - \frac{1}{n})^m.
\]
That’s the same answer we got by using indicators, but what a computation. Wow!

(b) There are \(\binom{a+b}{k}\) ways to choose \(k\) balls. If \(i\) are white, there are \(\binom{a}{i}\) to choose the \(i\) white balls and \(\binom{b}{k-i}\) ways to choose the remaining balls as black. So,
\[
P(i) = \binom{a}{i} \binom{b}{k-i}/\binom{a+b}{k}.
\]

**Exercise:** We know that \(\sum_{k=1}^{n} P(k) = 1\) since it is a PDF. Prove it.
To prove that \(\sum_{k} P(k) = 1\) you need a famous identity known as Vandermonde convolution, which is an application of the Binomial theorem:
\[
\sum_{k=0}^{a+b} \binom{a+b}{k} x^k = (1 + x)^{a+b} = (1 + x)^a (1 + x)^b
\]
\[
= \sum_{i=0}^{a} \binom{a}{i} x^i \sum_{j=0}^{b} \binom{b}{j} x^j
\]
\[
= \sum_{i=0}^{a} \sum_{j=0}^{b} \binom{a}{i} \binom{b}{j} x^{i+j}
\]
\[
= \sum_{k=0}^{a+b} \binom{a+b}{k} x^k = \sum_{i=0}^{a} \sum_{j=0}^{b} \binom{a+b}{k} x^k = \sum_{k=0}^{a+b} \alpha_k x^k
\]
On the left is a polynomial in \(x\). On the right is also a polynomial in \(x\). Two polynomials are equal if and only if their coefficients match. That is, \(\alpha_k = \binom{a+b}{k}\).

**Lemma 30.8** (Vandermonde convolution). \(\sum_{i=0}^{a} \binom{a}{i} \binom{b}{k-i} = \binom{a+b}{k}\).
Now for the expectation. We need \( \sum_{i=0}^{a} i P(i) = \sum_{i=0}^{a} i \binom{a}{i} \binom{b}{k-i} / (a+b)_k \):

\[
\sum_{i=0}^{a} i \binom{a}{i} \binom{b}{k-i} = \sum_{i=1}^{a} \binom{a}{i} \binom{b}{k-i} \quad (i = 0 \text{ term is zero})
= \sum_{i=1}^{a} \binom{a-1}{i-1} \binom{b}{k-i} \quad \text{use: } \binom{a}{i} = \binom{a-1}{i-1}
= a \sum_{i=0}^{a-1} \binom{a-1}{i} \binom{b}{k-i} \quad (\text{sum over } i = 0, \ldots, a-1)
= a \binom{a+b-1}{k-1}. \quad (\text{Vandermonde convolution})
\]

Finally, \( E[X] = a \binom{a+b-1}{k-1} / (a+b)^k \), as we got before with indicators.

**Chapter 21**

**Pop Quiz 21.1.** \( E[\Delta] = \frac{5}{36} + \frac{4}{18} + \frac{3}{12} + \frac{2}{9} + \frac{1}{6} + \frac{1\cdot 5}{36} + \frac{1\cdot 3}{9} + \frac{3}{12} + \frac{4}{9} + \frac{5}{18} = 0 \). In general,

\[ E[\Delta] = E[X - \mu] = E[X] - E[\mu] = \mu - \mu = 0. \]

**Exercise 21.2.**

(a) To construct the table on the right, we use \( E[X] = \frac{31}{2} \). From the table,

\[
\begin{align*}
\Delta^2 & = E[\Delta^2] = \frac{1}{36} \cdot (25 + 9 + 1 + 9 + 25) = \frac{35}{18} \quad \text{and} \quad P_X, \\
\sigma^2 & = \frac{\Delta^2}{E[\Delta^2]} = \frac{\Delta^2}{E[\Delta^2]}. \quad \text{std. deviation} = \sqrt{35/12}.
\end{align*}
\]

(b) We show the possible outcomes for the average of two dice, along with the PDF. We know that \( E[\text{average}] = \frac{31}{2} \).

\[
\begin{array}{c|ccccccc}
& 1 & 2 & 3 & 4 & 5 & 6 \\
\hline
X & 1 & 2 & 3 & 4 & 5 & 6 \\
\Delta^2 & \frac{25}{4} & \frac{16}{4} & \frac{9}{4} & \frac{4}{4} & \frac{1}{4} & \frac{4}{4} \\
P_X & \frac{1}{36} & \frac{1}{18} & \frac{1}{12} & \frac{1}{9} & \frac{5}{36} & \frac{1}{6} & \frac{5}{36} & \frac{1}{12} & \frac{1}{18} & \frac{1}{36}
\end{array}
\]

Note that the probabilities do not change. It is just the random variable that changed.

\[
\sigma^2 = \frac{\Delta^2}{E[\Delta^2]} = \frac{1}{36} \cdot (25 + 32 + 27 + 16 + 5 + 0 + 5 + 14 + 27 + 32 + 25) = \frac{35}{24} \quad \text{and} \quad \text{std. deviation} = \sqrt{35/24}.
\]

(c) The mean is \( p \), so the deviations are

\[
\Delta = \begin{cases} 
-p & \text{prob} = 1-p; \\
1-p & \text{prob} = p.
\end{cases}
\]

The standard deviation is \( \sigma = \sqrt{p(1-p)} \).

(d) \( \mu = 7 \) and \( \sigma = 2.52 \) so the event of interest is \( 5 \leq X \leq 9 \). So,

\[
P[\mu - \sigma \leq X \leq \mu + \sigma] = \frac{1}{6} + \frac{5}{36} + \frac{1}{6} + \frac{5}{36} + \frac{1}{6} = \frac{2}{3}.
\]

**Exercise 21.3.**

(a) \( E[X^2] = \frac{1}{4} (1^2 + \cdots + 6^2) = 15 \frac{1}{2} ; \ E[X^2] - E[X]^2 = 15 \frac{1}{2} - (3 \frac{1}{2})^2 = \frac{35}{12} \)

(b) \( E[X^2] = \frac{2^2 + 3^2 + 4^2 + \cdots + 6^2}{4 \cdot 36} = \frac{22}{4} \frac{1}{2} ; \ E[X^2] - E[X]^2 = \frac{22}{4} \frac{1}{2} + \frac{49}{4} - \frac{35}{24} = \frac{35}{24} \)

(c) \( E[X^2] = (1-p) \times 0^2 + p \times 1^2 = p; \ E[X^2] - E[X]^2 = p - p^2 = p(1-p) \)

**Exercise 21.4.** By linearity, \( E[Y] = E[a + bX] = a + b E[X] \). The deviations in \( Y \) are

\[
\Delta_Y = Y - E[Y] = a + bX - a + b E[X] = b(X - E[X]) = b \Delta_X.
\]

\[
\sigma^2(Y) = E[\Delta_Y^2] = b^2 E[\Delta_X^2] = b^2 \sigma^2(X).
\]

Let \( X \) be a Bernoulli. Then \( Y = 2X - 1 \). By Theorem 21.3, \( \sigma^2(Y) = 2^2 \sigma^2(X) = 4p(1-p) \).
Exercise 21.5. This exercise requires careful manipulation of sums that are squared.

\[
\sigma^2 \left( \sum_{i=1}^{n} a_i X_i \right) = E \left[ \left( \sum_{i=1}^{n} a_i X_i \right)^2 \right] - E \left[ \sum_{i=1}^{n} a_i X_i \right]^2 \quad \text{(definition)}
\]

\[
= E \left[ \sum_{i=1}^{n} a_i X_i \sum_{j=1}^{n} a_j X_j \right] - \left( \sum_{i=1}^{n} a_i E[X_i] \right)^2 \quad \text{(linearity)}
\]

\[
= \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j E[X_i, X_j] - \sum_{i=1}^{n} \sum_{j=1}^{n} a_i a_j E[X_i] E[X_j] \quad \text{(linearity)}
\]

\[
= \sum_{i=1}^{n} \sigma^2(a_i) E[X_i^2] - E[X_i]^2 - \sum_{i=1}^{n} \sum_{j \neq 1}^{n} a_i a_j (E[X_i] E[X_j] - E[X_i] E[X_j]) \quad \text{for independent random variables}
\]

The key step in the derivation above is to break the double sum which arises from squaring into the terms where \( i = j \) (\( X_i X_j = X_i^2 \)) and \( i \neq j \) (\( X_i \) and \( X_j \) are independent).

Exercise 21.6. \( X = X_1 + X_2 + X_3 + X_4 + X_5 \), where \( X_i \) are independent waiting times each with variance \( (1-p)/p^2 = 6 \). So \( \sigma^2(X) = 5 \times 6 = 30 \).

A Monte Carlo with 100,000 experiments gave an average wait time of 14.98 and a variance in the wait times of 29.99. A pretty good match to the theory.

Pop Quiz 21.7. \( X_1 = 1 \land X_2 = 0 \rightarrow X_3 = 0; \) \( X_1 = 0 \land X_2 = 0 \rightarrow X_3 = 0 \) or \( 1 \).

Exercise 21.8. \( X = \sum_{i=1}^{n} Y_i \) and

\[
E[X^2] = E \left[ \sum_{i=1}^{n} \sum_{j=1}^{n} X_i X_j \right] = \sum_{i=1}^{n} \sum_{j=1}^{n} E[X_i X_j] = \sum_{i=1}^{n} E[X_i^2] + \sum_{i=1}^{n} \sum_{j \neq 1}^{n} E[X_i X_j].
\]

(In the second step we used linearity of expectation.) \( X_i \) is a Bernoulli with probability \( p = \frac{1}{n} \) so \( E[X_i^2] = p \). \( X_i X_j \) is a Bernoulli with probability

\[
p = P[X_i = 1 \land X_j = 1] = \frac{(n-2)!}{n!} = \frac{1}{n(n-1)}.
\]

So, \( E[X_i X_j] = 1/n(n-1) \) and we have

\[
E[X^2] = \sum_{i=1}^{n} \frac{1}{n} + \sum_{i=1}^{n} \sum_{j \neq 1}^{n} \frac{1}{n(n-1)} = \frac{1}{n} + \frac{1}{n(n-1)} \times n(n-1) = 2.
\]

Since \( E[X] = 1 \), we have \( \sigma^2(X) = E[X^2] - E[X]^2 = 2 - 1 = 1 \).

Exercise 21.9.

(a) A fair dice roll is a \( U[6] \), so \( \sigma^2 = \frac{1}{12}(6^2 - 1) = 35/12 \). Since the \( n \) dice are independent, the variance of the sum is the sum of the variances and so \( \sigma^2(\text{sum of n dice}) = 35n/12 \).

(b) The expected sum is \( \mu(n) = 7n/2 \).

(c) Using (a) and (b),

\[
\mu \pm 3\sigma = 7n/2 \pm 3\sqrt{35n/12}.
\]

In the figure, the 3\( \sigma \)-envelope is gray. The max \( 6n \) and min \( n \) are dotted lines, and the mean \( \mu \) is the solid line.

(d) The bound is not trivial when \( 7n/2 + 3\sqrt{35n/12} \leq 6n \). Since \( n \) is an integer, this means \( n \geq 5 \).

Pop Quiz 21.10. Since \( X \) is a positive random variable, and \( E[X] = 1 \), \( P[X \geq 50] \leq \frac{1}{50} \).

Pop Quiz 21.11. Since \( \mu = 1 \) and \( \sigma = 1 \),

\[
P[X \geq 50] = P[X - 1 \geq 49] \leq P[|X - 1| \geq 49] = P[|X - \mu| \geq 49\sigma] \leq \frac{1}{49^2} \approx 0.00042.
\]

The first inequality is because \( X - 1 \geq 49 \rightarrow |X - 1| \geq 49 \); the second uses Chebyshev’s Inequality.
The probability that at least 50 men get the correct hat sums \( P(k) \) for \( k = 50 \) to 100.

\[
\mathbb{P}[X \geq 50] = \sum_{k=50}^{100} \frac{1}{k!} \sum_{i=0}^{100-k} \frac{(-1)^i}{i!} \approx 0.
\]

The probability is upper bounded by \( \frac{51}{50!} \) because, for \( k \in \{50, \ldots, 100\} \), \( \sum_{i=0}^{100-k} \frac{(-1)^i}{i!} \approx 1/e \) and \( 1/k! \leq 1/50! \). The exact value is much smaller than the Chebyshev or Markov bounds.

**Exercise 22.12.** \( X = X_1 + \cdots + X_n \), where \( X_i \) is Bernoulli with probability \( p = \frac{1}{2} \).

(a) By linearity, \( \mathbb{E}[X] = np = 100 \times \frac{1}{2} = 50 \). For the variance, \( \sigma^2(X_i) = p(1-p) \). By independence and linearity, \( \sigma^2(X) = np(1-p) = n \times \frac{1}{4} = 25 \) and \( \sigma = 5 \).

(b) Using the Binomial PDF, since \( X \) is a Binomial,

\[
\mathbb{P}[40 \leq X \leq 60] = \sum_{k=40}^{60} \binom{100}{k} \left( \frac{1}{2} \right)^k \left( \frac{1}{2} \right)^{100-k} = \frac{1}{2^{100}} \sum_{k=40}^{60} \binom{100}{k} \approx 0.9648.
\]

(c) \( \mathbb{P}[40 \leq X \leq 60] = \mathbb{P}[|X - \mu| < 10] = \mathbb{P}[|X - \mu| < 2\sigma] \geq 1 - \frac{1}{2^6} = 0.75 \).

**Exercise 22.13.**

(a) \( \mathbb{P}[|X - \mu| \geq t\sigma] = \mathbb{P}[X - \mu \geq t\sigma] + \mathbb{P}[X - \mu \leq -t\sigma] \). Using (21.2),

\[
\mathbb{P}[X - \mu \geq t\sigma] = \mathbb{P}[X \leq \mu + t\sigma] = \phi \left( \frac{\sqrt{n}(\mu - t\sigma)}{\sigma} \right) \approx \phi(-t\sqrt{n}) = 1 - \phi(t\sqrt{n}).
\]

\[
\mathbb{P}[X - \mu \leq -t\sigma] = 1 - \mathbb{P}[X - \mu \leq t\sigma] = 1 - \mathbb{P}(X = \mu + t\sigma) \approx 1 - \phi(t\sqrt{n}).
\]

Adding, these two equations gives the desired result: \( \mathbb{P}[|X - \mu| \geq t\sigma] \approx 2(1 - \phi(t\sqrt{n})) \).

(b) When \( t\sqrt{n} \) is large, \( \phi(t\sqrt{n}) \approx 1 - e^{-\frac{1}{2}t^2}/\sqrt{2\pi nt^2} \). Therefore, \( \mathbb{P}[|X - \mu| \geq t\sigma] \approx 2 \sqrt{\frac{2\pi nt^2}{e}} \).

**Chapter 22**

**Exercise 22.1.**

(a) \( 1 \rightarrow 2, 2 \rightarrow 3, 3 \rightarrow 4 \) is a bijection from \( A \) to \( B \) so \( |A| = |B| \).

(b) Suppose \( n \leq k \), then \( a_1 \rightarrow b_1 \) is an injection from \( A \) to \( B \), so \( |A| \leq |B| \).

Suppose \( |A| \leq |B| \), so there is an injection \( f : A \rightarrow B \). We prove by induction on \( n \) that \( n \leq k \). If \( n = 1 \), then \( B \) contains \( f(a_1) \), so \( k \geq 1 \). Suppose the claim holds for \( n \). Consider any set \( A \) with \( n+1 \) elements. Let \( f(a_{n+1}) = b_1 \). Relabel the elements of \( B \) so that \( b_1 \rightarrow b_k \) and \( b_k \rightarrow b_{k+1} \) (swap \( b_1 \) and \( b_k \)). Now, \( f \) maps \( a_{n+1} \rightarrow b_{k+1} \). If there was an element \( a_r \) which mapped to \( b_k \) under \( f \), \( a_r \) now maps to \( b_k \). Now remove \( b_k \) from \( B \) and \( a_{n+1} \) from \( A \). The new \( f \) is an injection from \( a_1, \ldots, a_n \) to \( b_1, \ldots, b_{k-1} \). By the induction hypothesis, \( n \leq k - 1 \) or \( n+1 \leq k \), as was to be proved.

(c) Suppose \( |A| \leq |B| \) and \( |B| \leq |A| \). By (b), \( |A| \leq |B| \rightarrow n \leq k \) and \( |B| \leq |A| \rightarrow n \leq k \) \( \implies n = k \).

Therefore \( a_1 \rightarrow b_1 \) is a bijection from \( A \) to \( B \), which means \( |A| = |B| \).

(d) If \( A \subseteq B \), then for \( a \in A \), \( f(a) = a \) is an injection from \( A \) to \( B \). Therefore, \( |A| \leq |B| \).

(e) No: in (a), \( A \subseteq B \) and \( B \subseteq A \). If any two sets are comparable using a relationship, the relationship is a total order. The subset comparison does not give a total order. You can always compare two sets using the injection relationship. Either \( A \subseteq B \) or \( B \subseteq A \), so either \( |A| \leq |B| \) or \( |B| \leq |A| \), which means size comparison using injection gives a total order.

**Exercise 22.2.** First we show \( f \) is 1-to-1. Suppose not. Let \( n_1 \neq n_2 \) and \( f(n_1) = f(n_2) \). So,

\[
\frac{1}{2}(1 + (-1)^{n_1}(2n_1 - 1)) = \frac{1}{2}(1 + (-1)^{n_2}(2n_2 - 1)) \rightarrow (-1)^{n_1}(2n_1 - 1) = (-1)^{n_2}(2n_2 - 1).
\]

The sign of both sides must be the same, so \( (-1)^{n_1} = (-1)^{n_2} \) and we conclude \( 2n_1 - 1 = 2n_2 - 1 \). That is, \( n_1 = n_2 \), a contradiction. So, \( f \) is an injection.

Now we show that \( f \) is onto. Given \( z \in \mathbb{Z} \), we must find \( n \) for which \( f(n) = z \).

\[
z > 0 : n = 2z \rightarrow f(n) = \frac{1}{2}(1 + (-1)^{2z}(4z - 1)) = z;
\]

\[
z \leq 0 : n = 2|z| + 1 \rightarrow f(n) = \frac{1}{2}(1 + (-1)^{2|z|+1}(4|z| + 1)) = -|z| = z.
\]

Therefore, \( f \) is onto, and hence a bijection from \( \mathbb{N} \) to \( \mathbb{Z} \).

**Pop Quiz 22.3.** \( f \) is an injection, so two elements of \( A \) cannot map to the same element of \( \mathbb{N} \).

**Pop Quiz 22.4.** Yes (every integer has a unique list position). Mathematically, one can show \( z_1 \neq z_2 \rightarrow f(z_1) \neq f(z_2) \). The positions of \( \{0, +3, -3, +6, -6\} \) are \( \{1, 6, 7, 12, 13\} \).

**Exercise 22.5.**
(a) The zig-zag path moves out one square at a time. Given \( \frac{z}{n} \in \mathbb{Q} \), the column in which \( z \) appears is \( c = 2z \) if \( z > 0 \) and \( c = 2|z| + 1 \) if \( z \leq 0 \). We want to know when the path hits the \( i \)th column and \( n \)th row. This entry will be hit when traversing the column and row for the square of size \( \max(n, c) \). If \( z > 0 \) (even), you come down the RHS of the square; if \( z \leq 0 \) (odd), you go up the RHS of the square. So, there are four cases:

\[
\begin{align*}
&n \geq c; \ n \ even & i &= (n-1)^2 + 2n - c = n^2 - c + 1 \\
&n \geq c; \ n \ odd & i &= (n-1)^2 + c \\
&c > n; \ c \ even & i &= (c-1)^2 + n \\
&c > n; \ c \ odd & i &= (c-1)^2 + 2c - n = c^2 - n + 1
\end{align*}
\]

(b) What is the list position of \( \frac{9}{7} \)? All positions are used by rationals with denominator 1.

(c) The sets are countable, so they can be listed \( \{A_1, A_2, A_3, \ldots \} \). Each set’s elements are countable, so they too can be listed, for example: \( A_1 = \{A_{1,1}, A_{1,2}, A_{1,3}, \ldots \} \). Therefore, all the elements in \( A_1 \cup A_2 \cup A_3 \cup \cdots \) can be presented in a grid as follows (similar to \( \mathbb{Q} \)):

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>( \mathbb{N} )</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>( A_{1,1} \rightarrow A_{1,2} )</td>
<td>( A_{1,3} \rightarrow A_{1,4} )</td>
<td>( A_{1,5} \rightarrow A_{1,6} )</td>
<td>( A_{1,7} )</td>
<td>( \cdots )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>( A_{2,1} \leftarrow A_{2,2} )</td>
<td>( A_{2,3} )</td>
<td>( A_{2,4} )</td>
<td>( A_{2,5} )</td>
<td>( A_{2,6} )</td>
<td>( A_{2,7} )</td>
<td>( \cdots )</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>( A_{3,1} \rightarrow A_{3,2} )</td>
<td>( A_{3,3} )</td>
<td>( A_{3,4} )</td>
<td>( A_{3,5} )</td>
<td>( A_{3,6} )</td>
<td>( A_{3,7} )</td>
<td>( \cdots )</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>( A_{4,1} \leftarrow A_{4,2} \leftrightarrow A_{4,3} \leftrightarrow A_{4,4} \leftrightarrow A_{4,5} \leftrightarrow A_{4,6} \leftrightarrow A_{4,7} \leftrightarrow \cdots )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>( A_{5,1} \rightarrow A_{5,2} \rightarrow A_{5,3} \rightarrow A_{5,4} \rightarrow A_{5,5} \rightarrow A_{5,6} \rightarrow A_{5,7} \rightarrow \cdots )</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
|   | \( \vdots \) | \( \vdots \) | \( \vdots \) | \( \vdots \) | \( \vdots \) | \( \vdots \) | \( \ddots \)

The path of arrows starting at \( A_{1,1} \) lists the elements in the union. So, the union is countable.

(d) We have not shown \( \mathbb{R} \) is countable (it’s not), so we don’t know if we can list the columns.

**Exercise 22.6.**

(a) No: \( b \), the complement of the diagonal, is infinite, and does not have to be in the list.

(b) Countable means you must produce the list (injection to \( \mathbb{N} \)) and show that it is a valid injection. Cantor diagonalization shows that there is no valid injection from \( B_\infty \) to \( \mathbb{N} \). Adding \( b \) to the list won’t help because the complement of the new injection will not be in the list.

(c) Every infinite binary string is a subset of \( \mathbb{N} \). The ones in the string identify the elements of \( \mathbb{N} \).

This is a bijection between the subsets and infinite binary strings, so |subsets of \( \mathbb{N} \)\| = |\( B_\infty \)\|.

(d) Given a finite subset, construct a finite binary string for the subset by taking its infinite binary string and truncating it at the last 1. For example \( \{1, 3, 7\} \mapsto 1010001 \). Two different subsets will have 1’s in different positions and so the truncated finite binary strings will be different. We have an injection from finite binary strings to \( B \) which proves that |\{finite binary strings\}| \leq |\( B \)| \leq |\( \mathbb{N} \)|. So, the finite binary strings are countable.

**Exercise 22.7.**

(a) For any string that is eventually zero, construct a finite binary string by truncating it at the last 1. Two different strings which are eventually zero have 1s in different positions, and so will truncate to different finite strings. Hence, we have an injection from the strings which are eventually zero and finite binary strings. So,

\[ |\{\text{strings which are eventually zero}\}| \leq |\{\text{finite binary strings}\}| \leq |\mathbb{N}|. \]

(b) An infinite string is either eventually zero or it is not, so (by definition) \( B_\infty = B_0^\infty + B_\infty^\infty \).

(c) Suppose \( B_\infty^\infty \) is countable. In (a) we showed that \( B_0^\infty \) is countable. This means \( B_\infty = B_0^\infty \cup B_\infty^\infty \) is the union of two countable sets and, by Theorem 22.2, is countable. That contradicts Theorem 22.5. We conclude that \( B_\infty^\infty \) is uncountable.

(d) Let \( a_1a_2a_3 \cdots \) and \( b_1b_2b_1 \cdots \) be two different infinite strings in \( B_\infty^\infty \). We prove that they map to different values in \([0, 1]\) which proves the mapping is an injection and therefore \(|B_\infty^\infty| \leq |[0, 1]|\).
Let \( x_a = \sum_{i=1}^{\infty} a_i 2^{-i} \) and \( x_b = \sum_{i=1}^{\infty} b_i 2^{-i} \). Since \( a_i \neq b_i \) for all \( i \), by well ordering, there is some minimum \( k \) for which \( a_k \neq b_k \). Suppose \( a_k = 1 \) and \( b_k = 0 \). We have:
\[
x_a - x_b = 2^{-k} + \sum_{i=k+1}^{\infty} a_i 2^{-i} - \sum_{i=k+1}^{\infty} b_i 2^{-i}.
\]
Since \( a \) is not eventually zero, the \( a_i \) for \( i \geq k+1 \) cannot all be zero, so \( \sum_{i=k+1}^{\infty} a_i 2^{-i} > 0 \). Setting \( b_i = 1 \) for all \( i \geq k+1 \) gives \( \sum_{i=k+1}^{\infty} b_i 2^{-i} \leq \sum_{i=k+1}^{\infty} 2^{-i} = 2^{-k} \). Therefore,
\[
x_a - x_b > 2^{-k} + 0 - 2^{-k} = 0
\]
and \( x_a \neq x_b \) as was to be shown. The same argument works if \( a_k = 0 \) and \( b_k = 1 \).

**Chapter 23**

**Pop Quiz 23.1.**
(a) \( \mathcal{L}_{\text{push}} \) contains odd numbers, or strings that end in 1.
(b) Start the string with a the state of the light (1 on, 0 for off). The rest of the string is the binary string encoding the number of pushes. \( \mathcal{L}_{\text{push}} \) contains strings whose first and last bit are different. \( \mathcal{L}_{\text{push}} = \{01, 10, 001, 110, \ldots \} \).

**Pop Quiz 23.2.**
(a) (i) Open. (ii) Closed. (iii) Closed (not a valid sequence of walk-on/walk off). (iv) Open. (v) Open. (vi) Closed. (vii) Closed (not a valid sequence).

(b) Strings with more 1’s than 0’s in which every prefix has at least as many 1’s as 0’s.

**Pop Quiz 23.3.** The graph represented by the string is shown on the right.

The distance between vertices 1 and 5 is three, so the answer is (NO).

**Exercise 23.4.**
(a) Starting from 1, you get your answer after \( D \) decision problems, for lengths 1, 2, \ldots, \( D \).
(b) You answer (NO) after length \( n \), the number of vertices in the network.
(c) Start with the lengths 1, 2, 4, \ldots, doubling each time. Suppose you get the first (YES) at \( 2^k \) which is \( k+1 \) decision questions. You now narrowed the distance to \( 2^{k+1} < D \leq 2^k \). Perform a binary search in the interval \([2^{k-1}, 2^k]\). The number of questions asked in the binary search is \( O(\log_2(\text{length of interval})) \). The length of the interval is \( 2^k - 2^{k-1} = 2^{k-1} \). So the total number of questions is at most \( k+1 + O(\log_2(2^{k-1})) \) \( \in O(k) \). Since \( 2^{k-1} < D \), it follows that \( k < 1 + \log_2 D \in O(\log_2 D) \) and you need only \( O(\log_2 D) \) questions.

**Pop Quiz 23.5.** The (YES)-set \( \subseteq \{\text{finite binary strings}\} \), so it’s countable and hence can be listed.

**Exercise 23.6.**
(a) \( \mathcal{L}_{\text{balanced}} = \{\varepsilon, 01, 0011, 0101, 0110, 1001, 1010, 1100, \ldots\} \)
(b) The finite binary strings can be listed, \( \Sigma^* = \{w_1, w_2, \ldots\} \). A computing problem \( \mathcal{L} \) is a subset of \( \Sigma^* \), which can be identified by an infinite binary sequence, where the 1’s in the sequence identify the strings in the sequence identify the strings in \( \mathcal{L} \). We have an injection from infinite binary sequences (which are uncountable) to computing problems, so
\[
|\{\text{infinite binary sequences}\}| \leq |\{\text{computing problem}\}|
\]
Therefore, the computing problems are uncountable.
(c) \( \mathcal{L}_1, \mathcal{L}_1 \cup \mathcal{L}_2, \mathcal{L}_1 \cap \mathcal{L}_2 \) are all collections of finite binary strings, so they are computing problems.

**Pop Quiz 23.7.**
(a) (i) \( \{01, 011, 0111\} \).
(ii) \( \{\varepsilon, 00, 0000\} \).
(iii) \( \{00, 000, 001, 0000, 0001, 0010, 0011, 1000, 1001, 0100, 1000, 1100\} \).
(b) \( \{0, 1\}^* \) or *1.
(c) Here are two solutions: \( \{0\}^* \cdot \{\varepsilon, 1\} \cdot \{0\}^* \cdot \{\varepsilon, 1\} \cdot \{0\}^* \)
\[
((\{0\}^* \cup \{\{0\}^* \cdot 1\}) \cup \{\{0\}^* \cdot 1\}^* \cdot 1 \cdot \{0\}^*)
\]

**Pop Quiz 23.8.**
\[
\begin{align*}
\varepsilon & \rightarrow 00, 11 \rightarrow 0000, 1001, 0110, 1111 \\
0 & \rightarrow 000, 101 \rightarrow 0000, 1001, 0100, 1111 \\
1 & \rightarrow 010, 111 \rightarrow 0100, 1010, 0111, 1111
\end{align*}
\]

**Exercise 23.9.**
(a) \( L_{0^n1^k} = 0^*1^k \). We could not find a regular expression for \( L_{0^n1^n} \). The challenge is to enforce equality in the number of zeros and ones. If you find one, please let us know.

(b) We do not show the minimality clause which is there by default.

\[
\begin{align*}
L_{0^n1^k} : & \begin{cases}
\epsilon \in L_{0^n1^k} \quad & \text{[basis]} \\
w \in L_{0^n1^k} \Rightarrow 0 \cdot w \in L_{0^n1^k} \quad & \text{[constructor rules]} \\
w \in L_{0^n1^k} \Rightarrow w \cdot 1 \in L_{0^n1^k} \quad & \text{[constructor rules]}
\end{cases} \\
L_{0^n1^n} : & \begin{cases}
\epsilon \in L_{0^n1^n} \quad & \text{[basis]} \\
w \in L_{0^n1^n} \Rightarrow 0 \cdot w \cdot 1 \in L_{0^n1^n} \quad & \text{[constructor rules]}
\end{cases}
\]

Though there is no regular expression for \( L_{0^n1^n} \), the language is easy to describe recursively.

Pop Quiz 23.10.

(a) These strings are in \( L_{\text{push}} \). We show the path of states traversed, ending in the resting state.

(i) \( q_0 \cdot q_1 \cdot q_1 \cdot q_1 \).
(ii) \( q_0 \cdot q_0 \cdot q_0 \cdot q_0 \).
(iii) \( q_0 \cdot q_0 \cdot q_0 \cdot q_0 \).
(iv) \( q_0 \cdot q_0 \cdot q_0 \cdot q_0 \).

(b) None of these strings are in \( L_{\text{push}} \).

(i) \( q_0 \cdot q_0 \).
(ii) \( q_0 \cdot q_0 \cdot q_0 \).
(iii) \( q_0 \cdot q_0 \cdot q_0 \).
(iv) \( q_0 \cdot q_0 \cdot q_0 \).

Chapter 24

Pop Quiz 24.1.

(a) \( q_0 |> 0000 \) \( \leadsto \) \( q_1 |> 0000 \) \( \leadsto \) \( q_1 |> 0000 \).

(i) \( q_0 |> 1000 \) \( \leadsto \) \( q_2 |> 1000 \) \( \leadsto \) \( q_2 |> 1000 \).

(ii) \( q_0 |> 0001 \) \( \leadsto \) \( q_1 |> 0001 \) \( \leadsto \) \( q_1 |> 0001 \).

(iii) \( q_0 |> 0100 \) \( \leadsto \) \( q_2 |> 0100 \).

(b) Non-empty string containing only 0’s.

(c) (i) **YES** (the next transition is to \( q_1 \) and the machine stays there).

(ii) **YES** (the machine stays in \( q_1 \)).

(iii) **NO** (the machine cannot escape \( q_2 \)).

Exercise 24.2.

(a) When \( M \) processes a 1, it enters \( q_2 \) from which it never leaves, so if \( w \) contains a 1 then \( M(w) = \text{(NO)} \). If \( w = \epsilon \), then \( M \) stops in \( q_0 \) and rejects. If \( w = 0^n \) for \( n > 0 \), then \( M \) enters \( q_1 \) and never leaves, accepting. Therefore, \( L(M) = \{ 0^n | n > 0 \} \).

(b) (i) Strings with no 1’s: \( L(M) = \{ \epsilon \} \).

(ii) Strings which are not only 0’s: \( L(M) = \{ 0^n | n > 0 \} \).

(iii) Strings with an even number of zeros (including no 0’s and \( \epsilon \)).

(iv) Strings with an odd number of zeros.

(v) \( L(M) = \{ \epsilon, 0 \} \) (a finite language with just 2 **YES**-strings).

(vi) Every string except \( \epsilon \) and 0: \( L(M) = \{ 0^n | n > 0 \} \).

Exercise 24.3.

(a) We give a construction that generalizes to any finite language. Let \( \ell \) be the length of the longest string in \( L \). Construct the binary tree to depth \( \ell \) corresponding to every binary string of length at most \( \ell \), as shown in the DFA below. Every string of length at most \( \ell \) leads the DFA to its unique state, which is a **YES**-state or not depending on whether the string is in \( L \). In the automaton below, the **YES**-states are \( s_{00}, s_{000}, s_{101}, s_{111} \).
(b) One can generalize the construction in (a). Instead, we give a proof by induction on $\ell$, the length of the longest string in $L$. The base cases are the empty language $L = \emptyset$ and $L = \{\varepsilon\}$.

Suppose that any finite language with maximum string length at most $\ell$ can be solved by a DFA. Consider any language $L$ with maximum string length $\ell + 1$. $L$ has two types of strings: those starting with 0 and those starting with 1. So $L = (0 \cdot L_0) \cup (1 \cdot L_1)$ where $L_0$ contains the suffixes of strings in $L$ that start with 0 and $L_1$ the suffixes of the strings in $L$ that start with 1. $L_0$ and $L_1$ are finite languages with maximum string length at most $\ell$. By the induction hypothesis, there are DFAs $M_0$ and $M_1$ that solve $L_0$ and $L_1$ respectively. We now construct a DFA $M$ that solves $L$ as follows.

If a string starts with 0, the DFA transitions to $r_0$, the start state of $M_0$. The DFA then runs $M_0$ and accepts if and only the suffix is in $L_0$, i.e. if and only if the string is in $L$. The logic is similar for a string that starts with 1. So, $M$ accepts a nonempty string if and only if the string is in $L$. One final detail is the empty string. If $\varepsilon \in L$, simply make $q_0$ a yes-state.

Exercise 24.4.
(a) String in $L_1 \cap L_2$ must contain a zero and they must end in 1 so they must be of the form of a string containing 0 concatenated with a 1: $*0*1$.

(b) From the start state, you wait for 0, transition to $s_1$ and wait for 1. If you get 1, you must ensure it is the last bit. The DFA which implements this logic is on the right.

(c) Using product states, the DFA structure is exactly the same as for the union $L_1 \cup L_2$. However, the DFA should accept only the states $q_i s_j$ where both $q_i$ is a YES-state of $M_1$ and $s_j$ is a YES-state of $M_2$. The resulting DFA is on the right.

Pop Quiz 24.5.
(a) $L$ contains a string with a zero concatenated with a string ending in 1, which gives any string with a zero that ends with 1. That is, $*0*1$. So we can use the same DFA in Exercise 24.4(b).

(b) From Exercise 24.4(a), $L = L_1 \cap L_2$. We can use the same DFA from Exercise 24.4(c).

(c) False. Consider any two $L_1, L_2$ with $L_1 \cap L_2 = \emptyset$. For example, $L_1 = \{0\}$ and $L_2 = \{1\}$.

Pop Quiz 24.6. In $M_1$, every zero toggles between $q_0$ and $q_1$. An odd number of 0’s leaves you in $q_1$ which accepts.
Similarly, in $M''_2$, the DFA toggles between $q_0$ and $q_1$ for every bit.

**Exercise 24.7.**

(a) After processing 100, the automaton state is $q_0$ or $s_1$, and 101 remains to be processed.

(b) $\{q_0\} \triangleright 100101 \xrightarrow{M''} \{q_0\} \triangleright 0101$

At the end of the computation the non-deterministic automaton could be in either of the states $q_0$, $s_0$ or $s_1$.

(c) If $s_1$ is one of the possible ending states, then there is a computation path that must have gone through $q_1$ (prefix in $L_1$) and ended in $s_1$ without re-entering $q_0$ (suffix in $L_2$), so $M''$ should accept. In this case the decision is **YES**.

(d) As the hint suggests, construct a state for every non-empty subset of the states in $M''$. We will use the subset as the label for the state. The states are

$$
\begin{align*}
&\{q_0\}, \{q_1\}, \{s_0\}, \{s_1\} \\
&\{q_0, q_1\}, \{q_0, s_0\}, \{q_0, s_1\}, \{q_1, s_0\}, \{q_1, s_1\} \\
&\{q_0, q_1, s_0\}, \{q_0, q_1, s_1\}, \{q_0, s_0, s_1\}, \{q_1, s_0, s_1\}
\end{align*}
$$

For each “subset”-state, the DFA transitions to a subset of states for each bit by considering all the possible ending states starting from any of the possible starting states of the transition. Consider, for example, state $\{q_1, s_1\}$ and input bit 1. $q_1 \xrightarrow{1} \{q_1, s_1\}$ and $s_1 \xrightarrow{1} s_0$, so

$$\{q_1, s_1\} \xrightarrow{1} \{q_1, s_0, s_1\}.$$  

The DFA accepts whenever $s_1$ is in the set of possible states. The full automaton is on the right. Though there are 15 subset states, only 8 of them are reachable from the start state.

Our “principled” approach may not give the most efficient DFA. For example, states $\{q_1, s_1\}, \{q_0, s_0, s_1\}, \{q_1, s_0, s_1\}$ state $\{q_0, s_0\}$ can be connected directly to are all accepting and only transition amongst themselves, so they can all be merged into a single accepting state.

**Exercise 24.8.**

(a)

The automaton starts in $q_0$ and processes the bits 10, transitioning to state $q_0$ then $q_1$. At the next 0, there are two possible actions so the computation branches. Each time the automaton is in state $q_1$ the computation branches. At the end, there are three possible computation paths the automaton could have taken (3 final states).

For further clarity, we show the “computation-tree” on the input 100101.
(b) The idea here is to start with a DFA that solves \( L \) and convert it into a DFA that solves \( L^\ast \).

We use the method in Exercise 24.3 to construct a DFA for \( L \).

DFA for \( L = \{1, 10\} \)

\[
\begin{array}{ccc}
& 1 & 0, 1 \\
0 & \rightarrow & 0, 1 \\
1 & \rightarrow & 0, 1 \\
E & \rightarrow & 0, 1 \\
\end{array}
\]

To implement \( L^\ast \), when we reach an accept state of \( L \), we would like the option to restart the DFA (as well as continue with the current path): any accept state, on receiving input 0, should also transition to the state \( s_e \) would transition to on receiving input 0. Similarly, on receiving input 1, the DFA should also transition to the state \( s_e \) would transition to on receiving input 1. We get a non-deterministic automaton. So, for example, \( s_1 \) transitions to \( s_1 \) and \( e \) on input bits 1 and 0 respectively because those are the transitions from \( s_e \).

Let us emphasize that a non-deterministic automaton is an interesting machine in its own right. However, for our purposes, it merely serves as an intermediate tool for getting at the DFA we need. One more small detail: since \( \varepsilon \in L^\ast \), the non-deterministic automaton above only captures the non-empty strings in \( L^\ast \) (often denoted \( L^+ \)). At the end, we must augment the automaton to accept \( \varepsilon \). We now use the subset-state method to get the DFA for \( L^+ \). We only show the subset-states that are used in the DFA, not all 15 subset-states.

DFA for \( L^+ \)

\[
\begin{array}{ccc}
\{s_e\} & \rightarrow & \{s_1\} \\
\{s_1\} & \rightarrow & \{s_1, E\} \\
\{E\} & \rightarrow & \{s_10, E\} \\
\end{array}
\]

Lastly, to get the DFA for \( L^\ast \), simply make the start state \( \{s_e\} \) accepting.

(c) (i) \( \{0, 1, 00, 100, 0000, 1000, 110, 111, 010, 011, 1010, 1011, 1110, 0010, 0011, 0110\} \)

(ii) We use the same approach as in (b). From any accept state, we must allow the automaton to continue its current path or "restart" from state \( q_0 \) for the next bit. The only accept state is \( q_1 \), so \( q_1 \) should also transition to wherever \( q_0 \) transitions for bits 0,1. This gives the non-deterministic automaton for \( L(M)^+ \).

We now use subset-states to convert this non-deterministic automaton to a DFA for \( L(M)^\ast \). To accept \( \varepsilon \), we make the start state accepting. The result is:

DFA for \( L(M)^\ast \)

\[
\begin{array}{ccc}
\{q_0\} & \rightarrow & \{q_1\} \\
\{q_1\} & \rightarrow & \{q_0, q_1\} \\
\end{array}
\]

You will notice that this DFA accepts every string including \( \varepsilon \): \( L(M)^\ast = \Sigma^\ast \). This is no surprise because the Kleene star of any language that contains 0 and 1 is \( \Sigma^\ast \). So, a much simpler solution for \( L(M)^\ast \) is \( \rightarrow \varepsilon \). We are not after the simplest solution. We are after a systematic solution.

**Exercise 24.9.** Let \( L_1 \) contain strings whose number of 1’s is divisible by 2 and \( L_2 \) contain strings whose number of 1’s is divisible by 3. Let We first argue that \( L_1 \) and \( L_2 \) are regular, which means there are DFAs \( M_1 \) and \( M_2 \) to solve \( L_1 \) and \( L_2 \):
We can build $\mathcal{L}$ using set operations on $\mathcal{L}_1$ and $\mathcal{L}_2$: $\mathcal{L} = \mathcal{L}_1 \cup \overline{\mathcal{L}_2}$. By Theorem 24.2, $\mathcal{L}_1 \cup \overline{\mathcal{L}_2}$ is regular because $\mathcal{L}_1$ and $\mathcal{L}_2$ are regular (closed under union). Again, by Theorem 24.2, $\overline{\mathcal{L}_1 \cup \mathcal{L}_2}$ is regular (closed under complement). Therefore $\mathcal{L}$ is regular.

**Pop Quiz 24.10.**

**Pop Quiz 24.11.** The same proof used for Theorem 24.3 works here. If the DFA has $k$ states then for some $0 \leq i < j \leq k$, the DFA must give the same answer for $0^i1^j$ and $0^i1^j$, but the first is balanced and the second is not, which is a contradiction.

**Exercise 24.12.**

(a) The start state is $q_0$. It is an accept state if the input string is empty.

$q_0 \rightarrow e$: If the first bit is 1, transition to error, doing nothing to the stack.

$q_0 \rightarrow q_1$: If the first bit is 0, push it onto the stack and transition to $q_1$. (Step I.)

$e \rightarrow e$: remain in the error state for any input ($0, 1, \varepsilon$) and stack symbol ($0, 1, \varepsilon$), doing nothing to the stack. symbol. If the input is $\varepsilon$, the automaton stops and rejects.

$q_1 \rightarrow q_1$: if input is 0 stack is 0, push 0 onto the stack and remain in $q_1$. (Step II.)

(b) $\{0, 1, \varepsilon\} \rightarrow \{0, 1, \varepsilon\} \rightarrow \{\}$

The transitions $q_2 \rightarrow e$ are three separate instructions conveniently stacked on top of each other.

(c) Left of head-position is the substring processed and to its right the substring remaining. We also must show the stack which we do to the left of the current state (the top of the stack is rightmost in black). Let us denote our PDA by $M$. The traces for the four strings are:

\[
\begin{align*}
\emptyset | q_0 | &\xrightarrow{0} q_0 | 0 \rightarrow 0011 \varepsilon &\emptyset | q_0 | &\xrightarrow{\varepsilon} q_0 | 0 \rightarrow 0011 \varepsilon \\
\emptyset | q_2 | 10 \rightarrow 0 &\xrightarrow{0} q_1 | 0 \rightarrow 0 \varepsilon &\emptyset | q_2 | 10 \rightarrow 0 &\xrightarrow{\varepsilon} q_1 | 0 \rightarrow 0 \varepsilon \\
\emptyset | e | 010 \rightarrow &\xrightarrow{0} q_0 | 0 \rightarrow 0011 \varepsilon &\emptyset | e | 010 \rightarrow &\xrightarrow{\varepsilon} q_0 | 0 \rightarrow 0011 \varepsilon \\
\emptyset | e | 010 \rightarrow &\xrightarrow{0} q_1 | 0 \rightarrow 0 \varepsilon &\emptyset | e | 010 \rightarrow &\xrightarrow{\varepsilon} q_1 | 0 \rightarrow 0 \varepsilon \\
\text{STOP, REJECT} &\xrightarrow{0} q_0 | 0 \rightarrow 0011 \varepsilon &\text{STOP, REJECT} &\xrightarrow{\varepsilon} q_0 | 0 \rightarrow 0011 \varepsilon \\
\end{align*}
\]

(d) The PDA seems reasonably easy to implement mechanically. The only concern is that it requires an infinite stack to process inputs like $0^n1^m$ for arbitrarily large $n$. In practice, of course, we can only implement a PDA with a finite stack memory.

**Chapter 25**

**Pop Quiz 25.1.**

(a) $S \xrightarrow{0} T_0 A \xrightarrow{1} T_0 XT_1 \xrightarrow{1} T_0 T_0 T_1 T_1 \xrightarrow{1} T_0 0 T_1 T_1 \xrightarrow{1} 00 T_1 T_1 \xrightarrow{1} 001 T_1 \xrightarrow{1} 0011$.

(b) Cannot be derived; (c) Cannot be derived; (d) Cannot be derived;
The strings in this CFL are of the form $0^n1^n$ for $n \geq 0$.

**Pop Quiz 25.2.**

(a) (i) $S \rightarrow \text{phrase}\langle \text{verb}\rangle \rightarrow \text{phrase}\langle \text{noun}\rangle \text{runs}\langle \text{u}\rangle S$

(b) $A_i\langle \text{noun}\rangle \text{runs}\langle \text{u}\rangle S \rightarrow A_i\text{cat}\langle \text{noun}\rangle \text{runs}\langle \text{u}\rangle S$

(c) $A_i\text{cat}\langle \text{noun}\rangle \text{runs}\langle \text{u}\rangle \text{walks}$

(d) $A_i\text{cat}\langle \text{noun}\rangle \text{runs}\langle \text{u}\rangle \text{The}\langle \text{n}\rangle \text{walks}$

**Exercise 25.3.**

(a) (i) $S \rightarrow \langle \text{stmt}\rangle; S \rightarrow \langle \text{declare}\rangle; S \rightarrow \text{int}\langle \text{variable}\rangle; S \rightarrow \text{int}\langle \text{variable}\rangle; S \rightarrow \text{int}\langle \text{variable}\rangle; S$

(b) Only (i) is semantically correct. In (ii) the variable is used before it is declared. In (iii) one variable is declared and another is used.

(c) Add the white space variable in the rule for $S$ and a new rule to create the white space:

1: $S \rightarrow \langle \text{stmt}\rangle; WS \rightarrow \langle \text{stmt}\rangle$

2: $\langle \text{stmt}\rangle \rightarrow \langle \text{assign}\rangle \mid \langle \text{declare}\rangle$

3: $\langle \text{declare}\rangle \rightarrow \text{int}\langle \text{variable}\rangle$

4: $\langle \text{assign}\rangle \rightarrow \langle \text{variable}\rangle = \langle \text{integer}\rangle$

5: $\langle \text{integer}\rangle \rightarrow \langle \text{digit}\rangle \mid \langle \text{digit}\rangle 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

6: $\langle \text{digit}\rangle \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$

7: $\langle \text{variable}\rangle \rightarrow x \mid x\langle \text{variable}\rangle$

8: $W \rightarrow WW \mid \varepsilon \mid | \mid \n$

**Pop Quiz 25.4.** Let $w$ be a non-empty string in $L_{\text{equal}}$. Either $w$ starts with 0 or with 1. Consider the case $0, w = \varepsilon$. Since $w$ has an equal number of 0’s and 1’s, $w$ has more 1’s than 0’s. Therefore some prefix of $v$ (for example $v$ itself) has exactly more 1’s than 0’s. By the well ordering principle, there is a shortest prefix of $v$ that has more 1’s than 0’s. Call this prefix $v_1$. $v = v_1w_2$ and $v_1$ must end in 1 otherwise some shorter prefix has more 1’s than 0’s. Therefore, $v_1 = w_11$ and $w_1$ cannot have more 1’s than 0’s, since $v_1$ is the shortest prefix with this property. Therefore, $w_1$ must have an equal number of 1’s than 0’s, because otherwise $v_1$ would not have more 1’s. We have proved:

$$w = \varepsilon = 0v = 0v_1w_2 = 0w_1w_2$$

where $w_2$ has an equal number of 1’s and 0’s. Since $w$ has an equal number of 1’s and 0’s, this means that $w_2$ must also have an equal number of 1’s and 0’s. The case where $w$ starts with 1 uses identical reasoning.

**Exercise 25.5.**
30. Solutions to Quizes & Exercises

(a) (i) \( S \rightarrow \varepsilon \mid 1S \) (Every non-empty string in \( L \) is of the form \( 1w \) where \( w \in L \).)

(ii) By induction on the length of the derivation. The base cases are \( S \Rightarrow \varepsilon \) and \( S \Rightarrow 1 \). All other derivations start \( S \Rightarrow 1S \) followed by a shorter derivation for the \( S \) on the RHS. For this shorter derivation, by the induction hypothesis the string derived is \( 1^n \) for \( n \geq 0 \) and so the full derivation gives \( 1^{n+1} \).

(iii) By induction on \( n \): Suppose \( S \Rightarrow 1^n \). Then, \( S \Rightarrow 1S \Rightarrow 11^n \) is a derivation of \( 1^{n+1} \).

(b) (i) \( S \rightarrow A1 \)

\[ 2: A \rightarrow \varepsilon \mid 0 \mid 1 \mid AA \]

(ii) All derivations start \( S \Rightarrow A1 \) followed by a derivation from \( A \), therefore the final string is the string resulting from \( A \) followed by 1.

(iii) By induction on string-length, any string \( v \) can be derived from \( A \). The base cases are \( v = \varepsilon, 0, 1 \). If \( v = 0x \), then \( x \) is shorter and \( A \Rightarrow x \) (induction hypothesis), so \( A \Rightarrow AA \Rightarrow 0A \Rightarrow 0x \). Similarly if \( v = 1x \). Therefore, any string of the form \( v1 \) can be derived from \( S \):

\[ S \Rightarrow A1 \Rightarrow v1 \quad (\text{derivation of } v \text{ from } A). \]

(c) (i) \( S \rightarrow A00A \)

\[ 2: A \rightarrow \varepsilon \mid 0 \mid 1 \mid AA \]

(ii) All derivations start \( S \Rightarrow A00A \) followed by derivations from each \( A \), therefore the final string contains 00 because it is \( \varepsilon00 \) where \( A \Rightarrow v \) and \( A \Rightarrow w \).

(iii) In (b) we showed that any string can be derived from \( A \) so any string of the form \( \varepsilon00w \) can be derived from \( S \) by \( S \Rightarrow A00A \Rightarrow \varepsilon00w \) (derivations of \( v \) and \( w \) from the \( A \)'s).

(d) (i) \( S \rightarrow \varepsilon \mid 1S0 \)

(ii) Induction on the length of the derivation. Every derivation starts \( S \Rightarrow 1S0 \) followed by a shorter derivation on the RHS \( S \), which by the induction hypothesis gives \( 1^{n+1}0^n \). Therefore the full derivation gives \( 1^{n+1}0^n1^{n+1} \).

(iii) Induction on the length of the string. Suppose \( S \Rightarrow 1^n0^n \) then \( S \Rightarrow 1S0 \Rightarrow 11^{n+1}0^n0^n \) is a derivation of \( 1^{n+1}0^{n+3} \).

(e) (i) \( S \rightarrow AB \) \quad (S is composed of two types of strings, \( A \) and \( B \))

\[ 2: A \rightarrow \varepsilon \mid 0 \mid 1A0 \quad (A \text{ generates } 1^*0^k \text{ as in part (e)}) \]

\[ 3: B \rightarrow \varepsilon \mid 1B \quad (B \text{ generates } 1^* \text{ as in part (a)}) \]

(ii) \( S \) is a string derived from \( A \) followed by one derived from \( B \). In (e,ii) we showed that all derivations from \( A \) yield \( 1^*0^k \). In (a,ii) we showed that all derivations from \( B \) yield \( 1^* \), therefore all derivations from \( S \) yield \( 1^*0^k1^* \).

(iii) In (e,iii) we showed that every string of the form \( 1^*0^k \) can be derived from \( A \). In (a,iii) we showed that every string of the form \( 1^* \) can be derived from \( B \). Therefore every string of the form \( 1^*0^k1^* \) can be derived from \( S \).

(f) (i) The basic palindromes are \( \varepsilon, 0, 1 \). All other palindromes either start and end in 0 (with a palindrome in between) or they start and end in 1 (with a palindrome in between). This observation suggests the grammar

\[ 1: S \rightarrow \varepsilon \mid 0 \mid 1 \mid 0S0 \mid 1S1 \]

(ii) Induction on the length of the derivation. For the derivation, which starts 0S0 (the same logic applies if it starts 1S1), suppose ultimately that the middle \( S \) yields \( x \) (via a shorter derivation). By the induction hypothesis, \( x \) is palindrome, and so the final string 0x0 is a palindrome because \( (0x0)^n = 0x0^n = 0x0 \).

(iii) Induction on the length of the palindrome. Suppose \( w \) is a (non-basic) palindrome. Then \( w = 0x0 \) or \( w = 1x1 \) where \( x \) is a shorter palindrome. Therefore \( S \Rightarrow x \) by the induction hypothesis. Therefore \( S \Rightarrow 0S0 \Rightarrow 0x0 \) (or \( S \Rightarrow 1S1 \Rightarrow 1x1 \)) is a derivation of \( w \).

Exercise 25.6.

(a) *0*: \( 1: S_0 \rightarrow A0A \)

\[ 2: A \rightarrow \varepsilon \mid 0 \mid 1 \mid AA \]

*1*: \( 1: S_1 \rightarrow B1B \)

\[ 2: B \rightarrow \varepsilon \mid 0 \mid 1 \mid BB \]

For the union (strings containing 0 or 1):

\[ 1: S \rightarrow S_0 \mid S_1 \]

\[ 2: S_0 \rightarrow A0A \]

\[ 3: A \rightarrow \varepsilon \mid 0 \mid 1 \mid AA \]

\[ 4: S_1 \rightarrow B1B \]

\[ 5: B \rightarrow \varepsilon \mid 0 \mid 1 \mid BB \]
Let us walk through a systematic procedure for converting a CFG into Chomsky Normal Form. Our grammar is 1: S → ε | 0S1S | 1S0S. First, we make sure that the start variable does not appear on the RHS of any rule. To do this, we change the start variable and add a new rule from the new start variable to the old one:

1: S → X
2: X → ε | 0X1X | 1X0X

This is a trivial change which does not alter the strings which can be derived. All derivations simply start S → X and proceed as they would have before. Now remove all rules which take a variable to ε: if a variable A may transition to ε, remove that transition and replace every occurrence of the variable on the RHS of a rule with a new instance of the rule that replaces A with ε. Note that every instance of A must be replaced “independently”. For example X → 0X1X gets replaced by

X → 0X1X | 01X | 0X1 | 01

which captures all possible ways of replacing the X’s with ε. Our grammar becomes:

1: S → ε | X
2: X → 0X1X | 01X | 0X1 | 01 | 1X0X | 10X | 1X0 | 10

Now replace each terminal with a corresponding terminal variable (for example T₀ for 0
T₁ for 1) and add a rule from each terminal variable to its corresponding terminal:

1. S → ε | X
2. X → T₀X₁ | T₀T₁ | T₀T₁X₁ | T₀X₁T₁ | T₀X₁T₁T₀ | T₀T₁ | T₁T₀ | T₀T₁X₁ | T₁T₀ | T₁T₀
3. T₀ → 0
4. T₁ → 1

Every rule (except for the rules to terminals) now has the form

<variable> → string of <variables>.

Now perform variable reduction on the RHS of any rule that has more than two variables. One way to reduce the rule A₁ → A₂A₃ · · · Aₖ is

A₁ → A₂B₁; 
B₁ → A₃B₂; 
B₂ → A₄B₃; 
· · · 
B_{k−2} → A_{k−1}Aₖ

When you combine all the rules above, it becomes the single rule A₁ → A₂A₃ · · · Aₖ. In general, you can pick any consecutive variables on the RHS and reduce it to a new variable, while providing a new rule from the new variable to the pair. For our grammar let us use A₀ → T₀X and A₁ → T₁X, then the rule for X becomes

X → T₀T₁ | T₁T₀ | A₀A₁ | T₀A₁ | A₀T₁ | A₁A₀ | T₁A₀ | A₁T₀

Our grammar becomes

1. S → ε | X
2. X → T₀T₁ | T₁T₀ | A₀A₁ | T₀A₁ | A₀T₁ | A₁A₀ | T₁A₀ | A₁T₀
3. A₀ → T₀X
4. A₁ → T₁X
5. T₀ → 0
6. T₁ → 1

Finally, for rules that take a variable to a single variable (e.g. S → X), replace the variable on the RHS by the entire rule for that variable. Our grammar in Chomsky Normal Form is

1. S → ε | T₀T₁ | T₁T₀ | A₀A₁ | T₀A₁ | A₀T₁ | A₁A₀ | T₁A₀ | A₁T₀
2. X → T₀T₁ | T₁T₀ | A₀A₁ | T₀A₁ | A₀T₁ | A₁A₀ | T₁A₀ | A₁T₀
3. A₀ → T₀X
4. A₁ → T₁X
5. T₀ → 0
6. T₁ → 1

(b) Every application of a production rule increases the length of the hybrid string by 1 (starting from S of length 1). Every application of a terminal rule keeps the length of the hybrid string fixed. Since the final string has length n, this means there must be n − 1 steps which increase the length by 1 and n steps which convert variables to terminal for a total of 2n − 1 steps. This argument works because no rule can decrease the length of the hybrid string.

(c) For a grammar in Chomsky Normal Form and an input string w of length n, try all derivations of length 2n − 1. If w resulted from one of these (possibly exponentially many) derivations, then w ∈ L, otherwise w /∈ L. Note that this is a relatively inefficient procedure to test for membership in a CFL, but a finite one.

**Pop Quiz 25.8.** The final parse tree is boxed on the right of each derivation.

```
S ⇒ S /\ S ⇒ S /\ S ⇒ S /\ S ⇒ S /\ S ⇒ S /\ S ⇒ S /\ S ⇒ S /\ S ⇒ S /\ S ⇒ S /\ S ⇒ S /\ S ⇒ S /\ S ⇒ S /\ S ⇒ S /\  
S /\ S  
S /\ S  
S /\ S  
S /\ S  
S /\ S  
S /\ S  
S /\ S  
S /\ S  
S /\ S  
S /\ S  
```

Though the two derivations are different, the final parse trees are identical.

**Exercise 25.9.**

(a) Change the order of some of the transitions to get a different derivation.
We get a different derivation but the same parse tree.

At the end of both derivations, every variable transitions to 2 in some order. The derivation using the unambiguous grammar in (25.4) is much longer, the price one pays to obtain unambiguous parse trees (because additional information is embedded in the rules).

At the end of each derivation, every variable transitions to a terminal in some order. In case of ambiguity we identify in bold the variable that is transitioning.

(ii) To remove the ambiguity, we give just one way to add a 1 to the string, $S \rightarrow \varepsilon \mid 1S$.

Pop Quiz 25.10. $S \Rightarrow 0S0 \Rightarrow 01S10 \Rightarrow 011S110 \Rightarrow 011\#110$

Exercise 25.11.
Exercise 25.12.

(a) Yes. \(L_1\) is the concatenation of \(0^n1^n\) (context free, \(S \rightarrow \varepsilon \mid 0S1\)) with \(0^m\) (context free, \(S \rightarrow \varepsilon \mid 0S\)). Since CFLs are closed under concatenation, \(L_1\) is a CFL.

(b) Yes, for a similar reason. \(L_2\) is the concatenation of \(0^n\) with \(0^n1^n\).

(c) \(L_1 \cap L_2 = \{0^n1^n0^n \mid n \geq 0\} = L\).

(d) Suppose CFLs are closed under intersection. Then \(L = L_1 \cap L_2\) must be a CFL, which contradicts the fact that \(L\) is not a CFL. Therefore CFLs are not closed under intersection.

(e) Suppose CFLs are closed under complement. So, \(\overline{L_1}\) and \(\overline{L_2}\) are CFLs. CFLs are closed under union, so \(\overline{L_1} \cup \overline{L_2}\) is a CFL. This means \(\overline{L_1} \cup \overline{L_2}\) is a CFL (closed under complement). But

\[
\overline{L_1} \cup \overline{L_2} = \overline{L_1 \cap L_2} = L_1 \cap L_2 (\neq L).
\]

(The first step uses \(\overline{A \cap B} = \overline{A} \cup \overline{B}\).) We have proved that closure under union and complement implies closure under intersection. But CFLs are not closed under intersection; in particular, \(L\) is not a CFL (contradiction). Therefore, CFLs are not closed under complement.

Chapter 26

Pop Quiz 26.1. On 0110, the TM halts with \(\overline{\square}\) at step 1 because there is no \# in the input.
30. Solutions to Quizes & Exercises

Pop Quiz 26.2.

(a) \(q \rightarrow [0,0] \rightarrow r\) From \(q\) if you read 0, transition to \(r\), write 0 and move right.

(b) \(q \rightarrow [0,1] \rightarrow r\) From \(q\) if you read 0, transition to \(r\), and move right.

(c) \(q \rightarrow [\checkmark,0] \rightarrow r\) From \(q\) if you read marked 0, transition to \(r\).

(d) \(q \rightarrow [0,\checkmark] \rightarrow r\) From \(q\) if you read 0, mark it and transition to \(r\).

(e) \(q \rightarrow [0,\checkmark] \rightarrow r\) From \(q\) if you read 0, mark it, transition to \(r\) and move right.

(f) \(q \rightarrow [0,1,\checkmark] \rightarrow r\) From \(q\) if you read 0 or 1, write marked 1, transition to \(r\) and move left.

(g) \(q \rightarrow [0,\checkmark] \rightarrow r\) From \(q\) if marked, unmark transition to \(r\) and move left.

(h) \(q \rightarrow [\checkmark,0] \rightarrow r\) From \(q\) if marked, write 0 (unmarked), transition to \(r\) and move left.

(i) \(q \rightarrow [\checkmark,\checkmark] \rightarrow r\) From \(q\) if marked, transition to \(r\) and move left.

(j) \(q \rightarrow [0,\checkmark] \rightarrow r\) From \(q\) if 0, erase, transition to \(r\) and move left.

(k) \(q \rightarrow [\checkmark,\#] \rightarrow r\) From \(q\) if blank, write \#, transition to \(r\) and move left.

(l) \(q \rightarrow [\checkmark,\checkmark] \rightarrow r\) From \(q\) if marked, write marked 0 and transition to \(r\).

Pop Quiz 26.3. We simply identify the step 2 states in the automata for Step 1 and Step 2, and “snap” the automata together by merging the step 2 states. The result is:
Pop Quiz 26.4. At the end of each step the machine transitions to the success state for that step; this success step becomes the starting point for the next step. Merging the automaton from the previous pop quiz with the automaton for step 3 at the state $q_2$ gives

Merging the states $z_4$ and $o_4$ with the step-4 machine gives

Filling in step 5 gives the final Turing machine.
30. Solutions to Quizes & Exercises

(a) \(q_0 \rightarrow q_0 \rightarrow q_0 \rightarrow q_0 \rightarrow q_1 \rightarrow q_1 \rightarrow q_1\)  
\(\rightarrow \) step 2 \(\rightarrow\) step 2 \(\rightarrow \) step 2 \(\rightarrow \) step 2 \(\rightarrow \) step 2 \(\rightarrow\) step 2  
\(\rightarrow \) step 3 \(\rightarrow\) step 3 \(\rightarrow q_4 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_6 \rightarrow q_6 \rightarrow q_6 \rightarrow q_6 \rightarrow q_6 \rightarrow q_6 \rightarrow A\)

(b) \(q_0 \rightarrow q_0 \rightarrow q_0 \rightarrow q_0 \rightarrow q_1 \rightarrow q_1 \rightarrow q_1\)  
\(\rightarrow \) step 2 \(\rightarrow\) step 2 \(\rightarrow \) step 2 \(\rightarrow \) step 2 \(\rightarrow \) step 2 \(\rightarrow\) step 2  
\(\rightarrow \) step 3 \(\rightarrow\) step 3 \(\rightarrow q_4 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow E\)

(c) \(q_0 \rightarrow q_0 \rightarrow q_0 \rightarrow q_0 \rightarrow q_1 \rightarrow q_1 \rightarrow q_1\)  
\(\rightarrow \) step 2 \(\rightarrow\) step 2 \(\rightarrow \) step 2 \(\rightarrow \) step 2 \(\rightarrow \) step 2 \(\rightarrow\) step 2  
\(\rightarrow \) step 3 \(\rightarrow\) step 3 \(\rightarrow q_4 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow E\)

(d) \(q_0 \rightarrow q_0 \rightarrow q_0 \rightarrow q_0 \rightarrow q_1 \rightarrow q_1 \rightarrow q_1\)  
\(\rightarrow \) step 2 \(\rightarrow\) step 2 \(\rightarrow \) step 2 \(\rightarrow \) step 2 \(\rightarrow \) step 2 \(\rightarrow\) step 2  
\(\rightarrow \) step 3 \(\rightarrow\) step 3 \(\rightarrow q_4 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow q_5 \rightarrow E\)

Exercise 26.5. Step 1 and 2 are very similar to our Turing machine which solved \(w \# w\). Instead, we are now looking for two \#\#s and a specific format of 0’s and 1’s. A DFA solves this problem.

In Step 3 you are looking for the first unmarked 0; if you find it, mark it and move to \#: 

529
Note, in state \( q_3 \) we did not show what to do if the input is 1; and in state \( q_3 \) we did not show what to do for 1 and \( \_ \). This is because the TM already verified the input format so a 1 or \( \_ \) is not possible. Now for step 4.

(State \( q_1 \) is responsible for moving right unmarking the 1’s.) In step 6 you are just checking for no unmarked right 0. In step 5 you are moving right to mark a 0. We implement both steps together.

Exercise 26.6.
(a) Using a Turing machine to solve a regular language is like using a sledgehammer to hammer in a thumb-tack. Our TM is just a glorified DFA:
(b) There are two ways to solve the problem. The first is to insert a punctuation character between the two \(w\)'s and then use the Turing machine we already developed to solve the problem:

1. If the first symbol is \(\_\), ACCEPT (empty input).
2. Return to \(*\).
3. Move right to the first unmarked bit and mark it.
   - If you come to \(\_\) (equal number of unmarked bits right of \(\_\)):
     - mark it with \#
     - return to \(*\), unmarking any marked bits, and goto Step 5.
4. Move right to the first unmarked bit before \(\_\).
   - If none exists (odd-length string) REJECT.
   - Erase and copy the bit to the blank on its right;
     goto Step 2.
5. Use the Turing machine that solves \(w\#w\).
   ACCEPT or REJECT using the output of this Turing machine.

On the right we show the Turing machine in action up to the point where it invokes the Turing machine that solves \(w\#w\). The entire purpose of this Turing machine is to insert the punctuation character at the midpoint of the input, and can be viewed as a preprocessing machine which reconfigures the input into a format that be used by some other Turing machine. This concept should be familiar to computer scientists: using someone else's program to solve a task, but reconfiguring your input so that their program will accept the input.

Our second approach builds a Turing machine to directly solve \(ww\). The previous approach involves less effort since it leverages previously solved problems; but, this leads to program (Turing machine) bloat. Directly solving the problem gives a more compact Turing machine.
1: If the first symbol is \( \text{ } \), ACCEPT (empty input).
2: Return to \( \ast \).
   // Mark the first string with ✓ and the second with ✘.
3: Move right to the first unmarked bit and mark it ✓.
   If none exists (you come to ✘), GOTO Step 5.
4: Move right to the last unmarked bit and mark it ✘.
   If none exists (the first right symbol is \( \text{ } \) or ✘) rejects.
   (the input has an odd number of bits)
   Otherwise, after marking, GOTO Step 2.
After the loop involving steps 3 and 4, the input string is partitioned into two halves: the first is marked with ✓ and the second with ✘.
5: Return to \( \ast \).
   // Match each ✓-bit with a corresponding ✘-bit.
6: Move right to the first bit marked ✓.
   If none exists (you come to \( \text{ } \) or ✘) accepts.
   Otherwise remember the bit and unmark it.
7: Move right to the first bit marked ✘.
   If the bit does not match the bit remembered, reject.
   If it is a match, unmark the bit and GOTO Step 5.
It is now just a matter of constructing the machine-level instructions for each step and snapping them together to get the final Turing machine.
Let’s do it for our second approach which directly solves \( ww \).

In steps 1–3 above, the notation \( \ast \) means any symbol that is not ✘. Now for step 4 and 5.

Lastly, we must implement step 6 and 7 in which the bits marked ✓ are matched with bits marked ✘.
We encourage the reader to construct the Turing machine which first preprocesses the string into $w\#w$ and then cascade it with our Turing machine that solves $w\#w$.

**Pop Quiz 26.7.**

(a) (i) The Turing machine copies the bits over one by one.
1: Move right to the first $\bot$ and write $\#$.
2: Return to $\ast$.
3: Move right to first non-marked before $\#$.
   Remember and mark the bit.
   If you reach $\#$, return to $\ast$ unmarking all the $\checkmark$ and halt.
4: Move right to first $\bot$, write the remembered bit and goto step 2.

(ii) $L = \{ w\#w \mid w \in \Sigma^* \}$.

(b) (i) We use a $\times$ to simulate the punctuation $\#$.
1: Move right to the first $\bot$ and mark with $\times$.
2: Return to $\ast$.
3: Move right to first non-marked before $\times$.
   Remember and mark the bit with $\checkmark$.
   If you reach $\times$, unmark the bit, return to $\ast$ unmarking all the $\checkmark$ and halt.
4: Move right to first $\bot$, write the remembered bit and goto step 2.

(ii) $L = \{ww \mid w \in \Sigma^* \}$.

(c) (i) Write a 1 for every zero and repeat for every zero.
1: Move right to the first $\bot$ and mark with $\#$.

Exercise 26.8.

(a) (i) The Turing machine copies the bits over one by one.
1: Move right to the first $\bot$ and write $\#$.
2: Return to $\ast$.
3: Move right to first non-marked before $\#$.
   Remember and mark the bit.
   If you reach $\#$, return to $\ast$ unmarking all the $\checkmark$ and halt.
4: Move right to first $\bot$, write the remembered bit and goto step 2.

(ii) $L = \{ w\#w \mid w \in \Sigma^* \}$.

(b) (i) We use a $\times$ to simulate the punctuation $\#$.
1: Move right to the first $\bot$ and mark with $\times$.
2: Return to $\ast$.
3: Move right to first non-marked before $\times$.
   Remember and mark the bit with $\checkmark$.
   If you reach $\times$, unmark the bit, return to $\ast$ unmarking all the $\checkmark$ and halt.
4: Move right to first $\bot$, write the remembered bit and goto step 2.

(ii) $L = \{ww \mid w \in \Sigma^* \}$.

(c) (i) Write a 1 for every zero and repeat for every zero.
1: Move right to the first $\bot$ and mark with $\#$.
2. Return to $\ast$.
3. Move right to first non $\times$-marked 0 and mark with $\times$.
   
   If you reach $\#$, return to $\ast$ unmarking all 0’s and halt.
4. Return to $\ast$.
5. Move right to first non $\checkmark$-marked 0 and mark with $\checkmark$.
   
   If you reach $\#$, return to $\ast$ unmarking $\checkmark$’s (leaving the $\times$’s) and goto step 3.
6. Move right to first $\bigcup$ and write 0.
7. Move left to first $\checkmark$ and goto step 5.

(ii) $L = \{0^n #1^n 2^n \mid n \geq 0\}$.

(d) (i) Mark and replace the first with the last bit and vice versa and continue.
1. Move right to the first non-marked bit. Mark it and remember it.
   
   If you reach $\#$, return to $\ast$, erasing all marks and halt.
2. Move right to the last non-marked bit. 
   
   If there is none, return to $\ast$, erasing all marks and halt.
   
   Otherwise, remember it, replace it with the bit from step 1 and mark it.
3. Move left to the first marked bit.
   
   Replace the bit with the bit remembered in step 2 and goto step 1.

(ii) $L = \{w \# w r \mid w \in \Sigma^\ast\}$.

Pop Quiz 26.9. (a) $q_0 \rightarrow q_0 \rightarrow E$. 
(b) $q_0 \rightarrow q_1 \rightarrow A$.
(c) $q_0 \rightarrow q_0 \rightarrow q_1 \rightarrow q_0 \rightarrow q_1 \rightarrow q_0 \rightarrow q_1 \rightarrow \cdots$ (infinite loop).

Pop Quiz 26.10. Yes. Every decider also trivially recognizes its language so every decidable language is also recognizable.

Exercise 26.11. Let $M$ be Turing machine that decides $L$. We assume that the states, symbols and transition instructions are suitably punctuated.

1. Process each state to check that there is a valid instruction telling the machine what to do if in that state for every symbol.
2. For each state, marking it when you process it do 
3. For each symbol, marking it when you process it do 
4. Find the transition instruction that begins with the state and symbol and verify that it is a valid transition instruction.

Our Turing machine is verifying that the input string represents a valid Turing machine, much like one of the tasks of a compiler is to verify that the input program is a valid program.

Chapter 27

Pop Quiz 27.1.
(a) To grade the submission correctly, the TA must determine if the program prints “Hello World!”. 
   
   If the program does print “Hello World!”, Goldbach’s conjecture is false; or, if the program does not print “Hello World!”, Goldbach’s conjecture is true. In either case, the TA resolves an open conjecture and becomes famous.
(b) Run the ultimate debugger on the student’s submission. If the debugger says the program has no infinite loop, then Goldbach’s conjecture is false. If the program runs forever, Goldbach’s conjecture is true.
(c) When you run CONFUSE on CONFUSE, first CONFUSE runs AUTO-GRADE on CONFUSE.
   
   If AUTO-GRADE says NO (CONFUSE does not print “Hello World!”) then CONFUSE replaces the output file with “Hello World!” and halts. That is CONFUSE does print “Hello World!”: a contradiction.
   
   If AUTO-GRADE says YES (CONFUSE does print “Hello World!”) then CONFUSE erases the output file which means it does not print “Hello World”. Again, a contradiction.
   
   CONFUSE is not a valid decider since it cannot output an answer on some input. But CONFUSE exists if AUTO-GRADE exists, therefore AUTO-GRADE cannot exist. Something is FISHY here.
   
   If AUTO-GRADE does not exist, then what are the TA’s using to grade the CS1 assignments?

Exercise 27.2. When we mark a vertex, we also mark the edges containing this vertex. This requires a lot of zig-zagging (we supress the details). Note: the encoding $\langle G \rangle$ contains all the punctuation.
Marking node “1” in the edges requires zig-zagging. The TM cannot just “remember” the label “1” and mark in in the edges, because the TM cannot “remember” labels. The TM has a finite number of states with which to “remember” things. The vertex labels are arbitrary, and there can be arbitrarily many of them since the graph can have any finite size. Vertex label “1” is really “0000001” (ASCII code). What happens is that the TM marks the position of the vertex label and remembers the first bit in the label (marking it). Then the TM moves right to the edges looking for a vertex with that first bit. If it finds one, it marks the bit and then zig-zags to match and mark all bits. If all bits match, the vertex is marked as matched. If at some point the bit does not match, the vertex is marked as non-matched so that it does not have to be checked again. After processing the vertex, all edge vertices that were marked as non-matched are unmarked. In our figure above, we skipped over the steps when no match is found. At the end of this phase, the TM is ready to scan the edges for an unmarked vertex.

Exercise 27.3. No, because if \( w \not\in \mathcal{L}_1 \) but \( w \in \mathcal{L}_2 \), we need a TM \( M \) that will halt with accept. However, because \( M_1 \) is a recognizer for \( \mathcal{L}_1 \), in step 1 it may not halt which means the construction for \( M \) will not halt.
It is indeed the case that recognizable languages are closed under union, but we need a more sophisticated construction for $M$. Essentially $M$ must interleave the running of $M_1$ and $M_2$, so $M$ runs one step of $M_1$ and then one step of $M_2$ and so on. We do not give the details.

**Pop Quiz 27.4.** (See also the solution to Exercise 27.2.) The reason is that the TM can only remember finitely many things using its states. Since $M$ on the other hand can an arbitrary number of states, the TM cannot “remember” the state. It marks the state, and must match bit-by-bit for a match to the transition instruction (depending on the tape input at $\nabla$).

In order to match the state bit-by-bit, the TM zig-zags.

**Exercise 27.5.** This Demonic fixed TM is $U_{TM} : \mathcal{L}(U_{TM}) = \mathcal{L}_{TM}$, which we know is undecidable.

**Exercise 27.6.** Our sketch implements the program our diabolical student gave in Pop Quiz 27.1.

$$D_{tm} = \text{Diabolical student’s Turing Machine}$$

**INPUT:** empty tape.

1. Write 4 (in binary) on the tape.
2. Test if the number on the tape is a sum of two primes. If not, ACCEPT; otherwise add 2 to the number on tape and repeat step 2.

Running $A_{tm}$ on $D_{tm}$ with empty input solves Goldbach’s conjecture. Goldbach’s conjecture is false if and only if $A_{tm}$ accepts.

$A_{tm}$ is a general purpose tool that does not exist. That does not mean we cannot “mathematically” analyze the the special TM $D_{tm}$ to determine if it terminates successfully.

**Exercise 27.7.**

(a) $\mathcal{L}_{\text{empty}} = \{\langle M \rangle \mid M$ is a TM and $\mathcal{L}(M) = \emptyset \}$.

Suppose $E_{tm}$ decides $\mathcal{L}_{\text{empty}}$. We sketch a decider $H_{tm}$ for $\mathcal{L}_{\text{HALT}}$ that uses $E_{tm}$. Since $\mathcal{L}_{\text{HALT}}$ is undecidable, this is a contradiction, which proves that $E_{tm}$ does not exist.

$$H_{tm} = \text{Decider for } \mathcal{L}_{\text{HALT}} \text{ that uses } E_{tm}$$

**INPUT:** $\langle M \rangle\#w$, where $M$ is a TM and $w$ its input.

1. Modify $M$ to $M’$.

   **INPUT:** $w’$

   1. Run $M$ on $w’$.
   2. ACCEPT

2. Obtain $\langle M’ \rangle$, the encoding of $M’$.
3. Run $E_{tm}(\langle M’ \rangle)$ and ACCEPT if and only if $E_{tm}$ REJECTS.

There are several interesting points in our construction of $H_{tm}$.

(i) The inputs to $H_{tm}$ are $\langle M \rangle$ and $w$, and these are “hard-coded” into $M’$.

(ii) You can define another TM, in this case $M’$, inside a TM.

(iii) $M’$ accepts every input providing $M$ halts on $w$.

(iv) $H_{tm}$ never actually runs $M’$ or $M$ on $w$; it only encodes $M’$ into its description $\langle M’ \rangle$.

First $H_{tm}$ always halts, so it is a decider, because $E_{tm}$ is a decider and always halts. Second, $\mathcal{L}(M’) = \emptyset$ if and only if $M$ does not halt on $w$, therefore $H_{tm}$ is a decider for $\mathcal{L}_{\text{HALT}}$, which is the contradiction we desire.

(b) $\mathcal{L}_{\text{EQ}} = \{\langle M_1 \rangle\#\langle M_2 \rangle \mid M_1$ and $M_2$ are TM’s and $\mathcal{L}(M_1) = \mathcal{L}(M_2)\}$.

Suppose you have a decider for $\mathcal{L}_{\text{EQ}}$, call it $EQ_{TM}$. We construct a decider $E_{tm}$ for $\mathcal{L}_{\text{empty}}$ as follows. Let $M^*$ be a TM that immediately halts and rejects on every input. So $\mathcal{L}(M^*) = \emptyset$.

Now let $E_{tm}(\langle M \rangle) = EQ_{TM}(\langle M \rangle\#\langle M^* \rangle)$. $E_{tm}$ accepts if and only if $\mathcal{L}(M) = \mathcal{L}(M^*) = \emptyset$.

Therefore $E_{tm}$ decides $\mathcal{L}_{\text{HALT}}$, a contradiction, and so $EQ_{TM}$ does not exist.

**Exercise 27.8.** The dominoes must start with a domino with the same first character in the top and bottom. The only possibility is $d_1$,

$$d_1 = \begin{bmatrix} 10 & \text{True} \\ 101 & \text{False} \end{bmatrix}$$
For the next domino, the top first character must be 1 and the remaining characters on top must match characters on the bottom. The only possibility is $d_3$.

$$d_1d_3 = \begin{bmatrix} 10 & 101 \\ 101 & 011 \end{bmatrix} = \begin{bmatrix} 10101 \\ 101011 \end{bmatrix}$$

We still have a hanging 1 at the bottom. Repeating the argument, the only possibility is $d_3$ again, and again, and so on. You can never get rid of the hanging 1 at the bottom. The only possible solution is $d_1d_3d_3d_3\cdots$, which is infinite. That is, this instance of PCP has no solution.

**Exercise 27.9.** So now, each domino is described by $n, m$ where $n$ is the number of 1’s on top and $m$ is the number of 1’s on the bottom. So the input is $n_1, n_2, \ldots, n_k\#m_1, m_2, \ldots, m_l$.

We will just give the algorithm to decide this instance relying on the Church-Turing thesis which says this algorithm can be converted to a TM-decider. We encourage the reader with some spare time to actually construct the TM.

1: If $n_i > m_i$ for all $i$ or $n_i < m_i$ for all $i$ then **REJECT**.
2: **ACCEPT**

The algorithm is simple enough, testing to see if all the top values are either less than or greater than the bottom values. This algorithm always halts, and we must prove that the decision is always correct.

First suppose $n_i > m_i$ (resp. $n_i < m_i$) for all $i$. Then the top string will always be longer (resp. shorter) than the bottom string, and so it is not possible to have a match and **REJECT** is correct.

Now suppose it is not the case that $n_i > m_i$ for all $i$ or $n_i < m_i$ for all $i$. That is $n_i \leq m_i$ for at least one $i$ and $n_i \geq m_i$ for at least one $i$. There are two cases:

(i) If $n_i = m_i$ then $d_i$ is a trivial solution.

(ii) $n_i < m_i$ for at least one $i$ and $n_i > m_i$ for at least one $i$. Without loss of generality, we may assume $n_1 < m_1$ and $n_2 > m_2$. Consider the sequence of dominoes

$$d_1^{(n_2 - m_2)}d_2^{(m_1 - n_1)}.$$ 

The number of 1’s in the top is $n_1(n_2 - m_2) + n_2(m_1 - n_1)$; the number of 1’s in the bottom is $m_1(n_2 - m_2) + m_2(m_1 - n_1)$. The difference is

$$m_1(n_2 - m_2) + m_2(m_1 - n_1) - n_1(n_2 - m_2) - n_2(m_1 - n_1) = (m_1 - n_1)(n_2 - m_2) + (m_2 - n_2)(m_1 - n_1) = 0.$$ 

That is, the top and bottom strings match and **ACCEPT** is correct.

**Exercise 27.10.**

(a) Use a second tape as the counter. Every time the TM $M$ executes a transition instruction, it first moves right on the counter tape and writes 1, then it performs its instruction. So, for every instruction, the counter tape will have a 1. If a single tape TM is desired, merge the counter tape, move twice, each time erasing the tape (if you come to *), move R. All slots within $n$ of * are now erased.

(b) Each step touches at most one new tape-slot, so at most $n$ tape-slots are touched.

(c) First, come back to *. For every 1 on the counter move L one step. Now, for every 1 on the counter tape, move R twice, each time erasing the tape (if you come to *, move R). All slots within $n$ of * are now erased.

**Exercise 27.11.** The homework should require that the program should solve the problem within a given number of steps (CPU-cycles or time). It is possible to decide if a TM prints “Hello World!” and halts within a given number of steps. Simply simulate the machine for that number of steps. If it halts, examine the tape for “Hello World!”. If it does not halt, **REJECT**.

**Exercise 27.12.**

(a) Let $M$ and $\overline{M}$ be recognizers for $L$ and $\overline{L}$ respectively. We sketch a decider for $L$. First, here is a sketch that does not work.

$$D \equiv \text{Flawed Decider for } L \text{ that uses } M \text{ and } \overline{M}$$

**INPUT:** $w$ (the input to $M$ or $\overline{M}$)

1. Run $M$ on $w$; if $M$ **ACCEPTS**, then **ACCEPT**.
2. Run $\overline{M}$ on $w$; if $\overline{M}$ **ACCEPTS**, then **REJECT**.

We know that one of $M$, $\overline{M}$ must accept, because they are recognizers and either $w \in L$ or $w \in \overline{L}$. However, in step 1, $M$ may infinite-loop ($M$ is only a recognizer). In this case $D$ also
30. Solutions to Quizes & Exercises

infinite-loops and so it is not a decider, which must always halt. The solution is to interleave the running of $M$ with $\overline{M}$, and if either accepts at any point in the interleaved execution, then perform the appropriate action.

$$D = \text{Decider for } L \text{ that uses } M \text{ and } \overline{M}$$

**INPUT:** $w$ (the input to $M$ or $\overline{M}$)

1. Run $M$ for one step on $w$; if $M$ accepts, then ACCEPT.
2. Run $\overline{M}$ for one step on $w$; if $\overline{M}$ accepts, then REJECT.
3. GOTO step 1.

The easiest way to implement this interleaving is using a second tape. Each time you switch machines, you switch tapes. Now, since one of $M$, $\overline{M}$ must halt, $D$ also halts. If $w \in L$, then $M$ halts and accepts, so $D$ also accepts; If $w \notin L$, then $\overline{M}$ halts and accepts, so $D$ rejects.

That is, $D$ is a decider for $L$ and $L$ is decidable.

(b) In (a) we proved:

$$\text{if } L \text{ and } \overline{L} \text{ are recognizable then } L \text{ is decidable.}$$

The contrapositive (which is equivalent) is:

$$\text{if } L \text{ is undecidable then it is not the case that } L \text{ and } \overline{L} \text{ are recognizable.}$$

This is exactly what was to be proved.

(c) $L_{\text{tm}}$ recognized by $U_{\text{tm}}$, $L_{\text{tm}}$ is undecidable, therefore by (b) $\overline{L}_{\text{tm}}$ is unrecognizable.

$L_{\text{hal}}$ is recognizable (just simulate the machine and accept if it halts). $L_{\text{hal}}$ is undecidable, therefore by (b) $\overline{L}_{\text{hal}}$ is unrecognizable.

**Fact:** with every undecidable problem $L$ comes at least one unrecognizable problem ($L$ or $\overline{L}$).

Chapter 28

Exercise 28.1.

(a) At each scan, an equal number of 0’s and 1’s are marked, so the total number of bits is always even, so the algorithm will not prematurely exit until all the 0’s, and correspondingly all the 1’s, are marked.

(b) We prove the claim by strong induction on $n$, the number of unmarked bits. It is easy to verify the base cases with $n = 1$. For the induction step, suppose the claim holds when the number of unmarked bits is at most $n$ and consider the claim for $n + 1$ unmarked bits. If $n + 1$ is odd, then there is nothing to prove.

Suppose $n + 1$ is even. The number of unmarked 0’s and 1’s are either both even or both odd.

We consider each case separately.

(i) Both even: Let the number of unmarked 0’s be $2k$ and the number of unmarked 1’s be $2\ell$, with $\ell \neq k$. After one scan, marking every other 0 and 1, there will be $k$ unmarked 0’s and $\ell$ unmarked 1’s, with $k \neq \ell$ and $k + \ell \leq n$. By the induction hypothesis, at some point there will be an odd number of unmarked bits and the algorithm will reject.

(ii) Both odd: Let the number of unmarked 0’s be $2k + 1$ and the number of unmarked 1’s be $2\ell + 1$, with $\ell \neq k$. After one scan, marking every other 0 and 1, there will be $k$ unmarked 0’s and $\ell$ unmarked 1’s, with $k \neq \ell$ and $k + \ell \leq n$. By the induction hypothesis, at some point there will be an odd number of unmarked bits and the algorithm will reject.

In both cases, an odd number of unmarked bits will occur and the claim holds for $n + 1$, so the claim holds for all $n \geq 1$.

(c) We give the sketch and analyze the runtime.

$$M = \text{Efficient Turing Machine that solves } \{0^n \# 1^n\}$$

**INPUT:** Binary string $w$.

1. Check that the input has the correct format and return to $\ast$.
2. Check that the number of unmarked bits is even.
   - If not, REJECT.
   - If there are no unmarked bits, ACCEPT.
3. Mark every other unmarked 0 and every other unmarked 1. GOTO step 2.
Fixing the size of the input, the worst case runtime is for $0^n \# 1^n$, because otherwise the algorithm exits early with a reject. For $0^n \# 1^n$, steps 1, 2 and 3 can all be accomplished in a single scan of the input, that is $\Theta(n)$ steps. Abusing notation a little, the runtime is given by

$$\text{number of times steps 2 and 3 are executed} \times \Theta(n).$$

If the number of unmarked zeros is $2k$ or $2k + 1$ before step 3 is executed, then the number of unmarked zeros after step 3 is $2k$ (see part (b)). That is, each time step 3 is executed, the number of unmarked 0’s drops by a factor of at least 2. If step 3 is executed $m$ times, then the number of unmarked 0’s is at most $n/2^m$, which will be less than one if $2^m > n$. If the number of unmarked 0’s is less than one, the machine exits. This means the machine exits after $[1 + \log_2 n]$ steps and the worst case runtime is

$$[1 + \log_2 n] \times \Theta(n) \in \Theta(n \log n).$$

**Pop Quiz 28.2.** $\log n$, $\sqrt{n}$, $n$, $n^2 \log n$ are all fast because they are upper bounded by $n^3$, and $n^3$ is fast with $\lambda = 8$, because $(2n)^3 \leq 8n^3$.

$(\log_2 n)\log_2 n$, $n^{\log_2 n}$, $2\sqrt{n}$, $2^n$ are all not-fast because for $n \geq 2^{20}$, they are all at least $(\log_2 n)^{\log_2 n}$, and we show, by contradiction, that $(\log_2 n)^{\log_2 n}$ is not fast. Suppose $(\log_2 n)^{\log_2 n}$ is fast. So there is a function $f(n)$ for which

$$(\log_2 n)^{\log_2 n} \leq f(n) \quad \text{and} \quad f(2n) \leq \lambda f(n).$$

Because $f$ is fast, $f(2^k) \leq \lambda f(2^k-1) \leq \lambda^2 f(2^{k-2}) \leq \cdots \leq \lambda^{k-1} f(2)$. Since $f$ is an upper bound, $f(2^k) \geq k^k$. Choose any $k > \max(\lambda, f(2))$. Then

$$f(2^k) \geq k^k > k^{k-1} \cdot k > \lambda^{k-1} f(2),$$

which contradicts $f(2^k) \leq \lambda^{k-1} f(2)$. Therefore, $(\log_2 n)^{\log_2 n}$ is not fast.

<table>
<thead>
<tr>
<th>Fast</th>
<th>$\log n$</th>
<th>$\sqrt{n}$</th>
<th>$n$</th>
<th>$n^2 \log n$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Not fast</td>
<td>$(\log_2 n)^{\log_2 n}$</td>
<td>$n^{\log_2 n}$</td>
<td>$2\sqrt{n}$</td>
<td>$2^n$</td>
</tr>
</tbody>
</table>

**Exercise 28.3.** Let $n = 2^k$, so $k = \log_2 n$.

$$f(2^k) = 2^k f(2^{k-1}) = 2^{2k-1} f(2^{k-2}) = \cdots = 2^{2k-1} \cdots 2 f(2^0) = 2^{k+2k+1} f(1) = 2^{k(k+1)/2} f(1) = (2^k)^{(k+1)/2} f(1) = n^{\log_2 n + \frac{1}{2}} f(1).$$

**Pop Quiz 28.4.** Our algorithm assumes the input does not contain $\uparrow$ symbols.

1: Move to the end of the input and mark the last symbol.
2: Shift all marked symbols 1 step right
   repeat, moving right, until you come to two blank symbols.
3: Move left to the first unmarked symbol.
   Mark the symbol and got to step 2.
   If you come to $\uparrow$, unmark all symbols.

We illustrate with input $0\#0$ on the right. For an input of size $n$, step 1 takes $\Theta(n)$ steps. In steps 2 and 3, for $k$ marked symbols, the work done is approximately $2k$ steps to shift the marks and $2k$ steps to come back to an unmarked symbol, for a total of about $4k$. The number of marked symbols goes from 1 to $n$, and so the runtime is of this loop is

$$\sum_{k=1}^{n} 4k = 2n(n + 1) \in \Theta(n^2).$$

The total run time is in $\Theta(n + n^2)$ which is in $\Theta(n^2)$.
Pop Quiz 28.5. If \( M_{two} \) has implemented \( t \) steps, the heads are at most \( 2t \) slots right of their beacon (\( \ast \) or \( \ddagger \)). So, move \( L \) until you find the beacon (at most \( 2t \) moves) and then move \( R \) until you find the head (another at most \( 2t \) moves). The total number of steps is at most \( 4t \in O(t) \).

We made the assumption that the head is always right of the beacon (start point). Alternatively, in \( M_{one} \), we can implement left and right boundary markers, \( LB \) and \( RB \), which mark the left most and right most points touched. You only need to update \( LB \) if you move \( L \) from \( LB \) and similarly update \( RB \) if you move \( R \) from \( RB \). Now, to find the head, move to one of the boundary markers and then in the other direction until you reach the head. Again, this would be in \( O(t) \) steps.

Exercise 28.6.

(a) By Pop Quiz 28.4, step 1 takes \( O(n^2) \) steps. All operations in \( M_{one} \) are constant time except for switch, which runs in \( O(n) \) steps because \( M_{two} \) runs in \( \Theta(n) \) steps (see Pop Quiz 28.5).

So, each execution of step 4 is requires \( O(n) \) steps, and step 4 is executed \( n \) times, which is \( O(n^2) \) steps. Similarly for step 5. So the total number of steps is in \( O(n^2) \).

(b) The proof is exactly analogous to the analysis in (a). Reconfiguring the tape into odd and even slots requires \( O(n^2) \) steps. Now, \( M_{one} \) simulates each step of \( M_{two} \), with the added complication of switch. In the worst case \( M_{one} \) must switch at every step of \( M_{two} \), which is at most \( t(n) \) switch’s. Each switch requires \( O(t(n)) \) steps (Pop Quiz 28.5), so the number of steps spent switching is \( t(n) \times O(t(n)) \in O(t(n)^2) \) (abuse of \( O \)-notation). Once the switch is done, \( M_{one} \) simply implements \( M_{two} \)’s instruction in \( O(1) \) steps, so it takes \( O(t(n)) \) steps to implement all \( M_{two} \)’s instructions.

The total runtime is in \( O(n^2 + t(n) + t(n)^2) \). Notice that if \( M_{two} \) is some trivial machine that immediately rejects, then the overhead to reconfigure the tape, \( n^2 \), dominates. Assuming that \( M_{two} \) is a non-trivial TM that at least examines its input, then \( t(n) \geq n \) in which case \( t(n)^2 \) dominates and \( n^2 + t(n) + t(n)^2 \in O(t(n)^2) \), completing the proof.

If you had a \( k \)-tape machine, the only difference in the analysis is that

(i) The overhead to reconfigure the tape into \( k \) interleaved tapes is \( kn^2 \).

(ii) The switch takes \( O(kt(n)) \) steps.

(iii) An instruction of \( M_k \) on \( M_{one} \) takes \( O(k) \) steps (e.g. move \( R \) becomes move \( k \) steps \( R \)).

The runtime is \( O(kn^2 + kt(n) + kt(n)^2) \in O(kt(n)^2) \in O(t(n)^2) \) (since \( k \) is a constant).

Exercise 28.7. To prove a problems is decidable, we must sketch a Turing Machine to it.

(a) The idea is to try all permutations of the vertices.

\[
M = \text{Decider for HAMILTONIAN-PATH}
\]

**INPUT:** Encoding of a graph, \( (G) \).

1: Check that the input has the correct graph-format and return to \( \ast \).

2: Using the list of vertices, list out every permutation of the vertices to the right of the graph input (with a punctuation character \( # \) to separate each permutation).

3: Process each permutation of the nodes: \( v_{i_1} v_{i_2} \cdots v_{i_n} \).
   For consecutive vertices \( v_{i_k} v_{i_{k+1}} \), check that \( (v_{i_k}, v_{i_{k+1}}) \) is an edge.
   (Identify \( v_{i_{k+1}} \) with \( v_{i_1} \).
   If every edge exists ACCEPT; otherwise process the next permutation.

4: If you didn’t accept for any permutation, REJECT

The algorithm has non-trivial steps, e.g. how do you list out every permutation of the vertices. If there is one vertex it is easy. If there are \( n \), you can solve the problem “recursively”: for each vertex \( v \), listing out the permutations of the other \( n - 1 \) vertices and prepend \( v \) to the front.

(b) The idea is to try all subsets of the vertices of size \( \lceil n/2 \rceil \).
Otherwise write the

2

(a) Try every possible factor from 2 to \( n - 1 \). If none work, then **REJECT**. Otherwise write the
factor that works and **ACCEPT**.

---

### Exercise 28.7.

(a) Try every possible factor from 2 to \( n - 1 \). If none work, then **REJECT**. Otherwise write the
factor that works and **ACCEPT**.

---

### 30. Solutions to Quizes & Exercises
(b) We are going to take a particularly dumb approach to this problem. First compute the unary representation of \( w_n \) and then run the algorithm from part (a) on this unary representation.

\[
M_{\text{binary}} = \text{Transducer to compute a factor of } n.
\]

INPUT: \( w_n, n \) in binary. Assume \( n \geq 2 \).
1. Compute \( w_n \) in unary and write \( 1^{w_n} \) to the left of \( \star \).
2. Run the TM \( M_{\text{unary}} \) on the left of \( \star \) (replacing \( L \leftrightarrow R \)).
3. If \( M_{\text{unary}} \) rejects, reject.
4. If \( M_{\text{unary}} \) accepts,
   Compute the binary \( w_p \) for the unary \( p \) (which is on the left).
   Copy \( w_p \) to the right of \( w_n \) and erase the tape left of \( \star \).

A non-trivial task in step 1 is to convert binary to unary. It is not a very hard task. Let \( w_n = b_{k-1}b_{k-2} \cdots b_1b_0 \) be the \( k \) bits of \( w_n \), where \( b_{k-1} = 1 \). Going from \( i = 0 \) to \( k - 1 \), if \( b_i = 1 \), you add \( 2^{i-1} \) 1’s to the unary representation. We encourage the reader to construct a TM for binary-to-unary conversion. The details are not essential to our argument.

Since \( w_n = b_{k-1}b_{k-2} \cdots b_1b_0 \) and \( b_{k-1} = 1 \), \( n \geq 2^{k-1} = 2^{\log_2 w_n - 1} \). We know that \( M_{\text{unary}} \) requires at least \( n \) steps to run, so the runtime of \( M_{\text{binary}} \) is at least the runtime of \( M_{\text{unary}} \) on \( 1^{w_n} \) which is at least \( 2^{\log_2 w_n - 1} \) (the additional steps like the binary to unary conversion can only add more to the runtime). The runtime is at least exponential, which is non-polynomial.

We emphasize the difference between the natural parameter in the problem, \( n \), and the length of the input to the TM. Runtimes refer to the length of the input.

In (a), the natural parameter and the length of the input are comparable, equal to \( n \) and so a polynomial TM will have runtime which is polynomial in \( n \) (the length of the input).

In (b), the input is \( w_n \), which is the binary representation of \( n \). This input has length \( \log_2 n \), and so a polynomial TM for the problem with the input formatted in binary must have a runtime that is polynomial in \( \log_2 n \) (for example \( \log_2^3 n \) would work). Our runtime in (b) is at least \( 2^{\log_2 n - 1} \) which is not polynomial in \( \log_2 n \).

You may also be astounded by the simplicity of our algorithm in (b). Yes, it is stupid, but not far off from reality. Nobody has been able to find a factoring-algorithm that is a polynomial in \( \log_2 n \). So our algorithm which looks stupid is not that far off from the best we know how to do. Be the first to find an algorithm which is polynomial in \( \log_2 n \) and fame will come.

Chapter 29

Pop Quiz 29.1.
(a) (YES): For \( A = \{5, 11\} \), \( \text{sum}(A) = 16 \geq 15 = \frac{1}{2} \text{sum}(S) \)
(b) (NO): It is not obvious how to prove the answer is no. The simplest “proof” is to list out all the possible subset-sums. We used the simple program on the right to create this list of possible subset-sums. The idea is to start with \( 0 \), and add each value in \( S \) to the current possible subset-sums (removing duplicates). The possible subset-sums are:

\[
0, 2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 27, 28, 30.
\]

Observe that 15 is not one of the possible subset-sums.
(c) (YES): \( \text{sum}(S) = 30 \) and \( 3 + 6 + 6 = 15 \) (also \( 2 + 11 + 2 = 15 \)).

Exercise 29.2.
(a) 001101.
(b) The evidence would be an encoding of all subsets. The certifier would check each subset’s sum to verify that it does not work. Since there are exponentially many subsets, the evidence is exponential and the runtime is exponential.
(c) Let \( n \) be the number of elements in the input \( S \). Assume that the runtime of the certifier (given the input) is a polynomial \( p(n) \). To solve a problem with input \( S \), run the certifier
with every possible setting for the certificate (n-bit string). If any certificate verifies to $$\text{YES}$$ then accept. If no certificate verifies to $$\text{YES}$$ then reject.

There are $$2^n$$ possible certificates, so the runtime is $$2^n p(n)$$, which is not polynomial. (Technically, $$n$$ is not the length of the input, but the input length is related to $$n$$.)

**Exercise 29.3.**

(a) Easier to prove $$\text{YES}$$. The evidence is the subset; the proof checks that the subset-sum is $$k$$.

(b) Easier to prove $$\text{YES}$$. The evidence is the clique; the proof checks that there are at least $$k$$ clique-vertices and there is an edge between every pair of vertices in the clique.

(c) Easier to prove $$\text{YES}$$. The evidence is an assignment of colors to every vertex; the proof checks that there are at most $$k$$ different colors and that every edge has two vertices of different colors.

(d) Easier to prove $$\text{NO}$$. The evidence is integers $$p, q$$; the proof checks $$p, q > 1$$ and $$pq = n$$.

The $$\text{YES}$$-answer can also be proved quickly because the AKS-test (proved in 2002) shows that IsPrime $$\in \mathbb{P}$$. If you can solve a problem in polynomial time, you can prove either answer in polynomial time. Still, it is much easier to prove the $$\text{NO}$$-answer.

(e) Easier to prove $$\text{YES}$$. The evidence is an isomorphism $$f$$ from $$G_1$$ to $$G_2$$; the proof checks that the number of vertices and edges match and for each edge $$(u, v) \in G_1$$, $$(f(u), f(v)) \in G_2$$.

**Pop Quiz 29.4.** The valid inputs have length $$\frac{1}{2} n(n - 1) = 0, 1, 3, 6, 10, 15, 21, 28, 36, \ldots$$

(a) 12 is not a valid input length.

(b) The algorithm above the number of 1’s on the second tape is $$2^{i-1}$$ when bit in $$k$$ being processed. The head on the main tape just moves right along the evidence $$e$$, marking 1’s, making at most $$n$$ steps in total. The expensive step is doubling the number of 1’s on the second tape. If there are $$2^{i-1}$$ 1’s then adding another $$2^{i-1}$$ 1’s takes $$\Theta((2^{i-1})^2)$$ steps. Let $$f$$ be the number of bits in $$k$$, then the work in doubling is in $$\Theta(\sum_{i=1}^{f} 2^{2^{i-2}}) = \Theta(\frac{1}{4}(4^f))$$. Since $$k \leq n$$, it means $$2^f \leq n$$ and so the total time spent in step 4 is in $$\Theta(n^2)$$. Assuming phase 1 is a success, in phase 2 the algorithm must ensure that an edge is present between every pair of vertices in the clique. Equivalently, for every edge not present, we ensure that pair of vertices are not both present in $$e$$. Our algorithm needs to mark each edge while simultaneously keeping track of which pair vertices in $$e$$ that edge connects.
On the right, we illustrate the algorithm keeping track of the node-pairs as it walks through the edges on
1: Mark the first edge \( e_1 \) and the first two vertices in \( E \).
2: If the last edge marked is 0, \text{REJECT} if both marked vertices are 1.
3: Mark the next edge. If there are no more, \text{ACCEPT}
4: Update the vertex marks and \text{GOTO} step 2.

To update the vertex marks, move the \( \checkmark \) one right.
If the \( \checkmark \) cannot move right,
move the \( \times \) one right and the \( \checkmark \) to the right of the \( \times \).

Walking through the edges one step at a time is \( \frac{1}{2} n (n - 1) \) steps. For each edge, you move right to the \( \checkmark \) and update it. Finding the \( \checkmark \) is \( O(|(G)|) \) since \(|(G)|\) dominates the size of the input, and updating is \( O(n) \). So, there are \( \frac{1}{2} n (n - 1) \) steps each taking \( O(n^2) \) time, for a total of \( O(n^3) \) time, which dominates the runtime of the entire algorithm.

The total runtime is in \( O(n^3) \). The input size is in \( \Theta(n^2) \), so we have a quadratic algorithm.

Exercise 29.6.
(a) \text{COLORING:} The evidence \( E \) is a set of colors, 1, 2, \ldots, \( \ell \) and a color assignment to each vertex.

Since \( \ell \leq n \), the encoding of each color is at most \( \log_2 \ell \) bits and the encoding of all the colors has length at most \( \ell \log_2 \ell \leq n \log_2 n \). The color assignments to each vertex also requires at most \( n \log_2 n \) bits. So the |\( E \)| is polynomial. The certificate checks that the number of colors \( \ell \) is at most \( k \). Then the certificate processes each edge \((u,v)\) and verifies that the colors assigned to \( u \) and \( v \) are different. The certificate is polynomial because it has polynomially many things to do, each taking polynomial time.

(b) Any problem in \( \mathsf{P} \) has a polynomial decider. The certificate is just this polynomial decider with no evidence \( E = \varepsilon \). Any polynomial time decider is a polynomial time verifier of \( \text{YES} \).

Pop Quiz 29.7. The second automaton is the first one with \( \text{YES} \) and \( \text{NO} \) states switched. Yet, strings accepted by the first automaton are not necessarily rejected by the second and \text{vice versa}.

<table>
<thead>
<tr>
<th>first automaton</th>
<th>second automaton</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \varepsilon )</td>
<td>( \varepsilon )</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>01</td>
<td>01</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
</tr>
</tbody>
</table>

To solve the complement language, you can’t just flip accept to reject states and \text{vice versa}. For example, a string is accepted if (say) 2 of 4 computation paths accept. Now, if you switch the \( \text{YES} \) and \( \text{NO} \) states, you still have 2 of 4 computation paths accepting, so you will accept.

There is asymmetry between accept and reject. To accept, \text{one} of the possible computation paths must accept. To reject, \text{every} possible computation path must reject. (Compare with the \( \text{YES} \)-\( \text{NO} \) asymmetry in defining \( \mathsf{NIP} \).) To solve the complement language, you first construct the equivalent DFA using subset states (Exercise 24.7), and then flip the \( \text{YES} \) and \( \text{NO} \) states of the DFA.

Pop Quiz 29.8. The deterministic instruction is: if in state \( q_1 \) and the bit read in is \( `0` \), then transition to state \( q_3 \) while writing \( `1` \) to the tape and then move \( \mathsf{R} \).

The non-deterministic instruction is: if in state \( q_1 \) and the bit read in is \( `0` \), then transition to one of the states \( q_0, q_1, q_3 \) (which?), either mark the \( `0` \) or replace it with \( `1` \) (which?), and move either \( \mathsf{R} \) or \( \mathsf{L} \) (which?).

The non-deterministic machine gets to try all of these choices, so there are \( 2 \times 2 \times 3 = 12 \) possible choices. The computation splits into 12 possible branches as shown on the right.

In keeping with the tradition for non-deterministic automata, the non-deterministic TM will accept if any one of these branches is an accepting computation.

Pop Quiz 29.9. \( a) \ y = (1 \land 0) \land (1 \land 1) = 0 \land 1 = 0. \) \( b) \ Y e s. \ x_1 x_2 = 01. \)

Exercise 29.10.
(a) Given input \( w \) and evidence \( E \) of length \( p(n) \), the certificate \( M \) runtime is \( t(n) \), a polynomial
in \( n \). By Theorem 29.3 in time \( \text{poly}(t(n)) \), we construct a circuit of size \( \text{poly}(t(n)) \) which is fed into the black-box. The runtime of the black-box is polynomial in the size of its input, therefore, the runtime of the black-box is \( \text{poly}(\text{poly}(t(n))) \). So,

\[
\text{total runtime} = \text{time to create circuit } + \text{time to run the black-box} = \text{poly}(t(n)) + \text{poly}(\text{poly}(t(n))),
\]

which is polynomial because a polynomial evaluated on a polynomial is a polynomial.

(b) We must construct different certifier circuits for each evidence length from 0, 1, \ldots, \( p(n) \). So, we need to run the black-box \( p(n) \) times, for each of the \( p(n) \) circuits, which essentially multiplies the polynomial running time by \( p(n) \). Since the product of polynomials is a polynomial, the runtime remains polynomial.

Note that if you wish to only use the black-box just once, you can take each of the \( p(n) \) circuits and send them through a massive \( -\text{way} \) black-box. Feed this gigantic circuit (which is only about \( n \) times bigger than the one in (a)) into the black-box. If the black-box says the circuit is satisfiable, then one of the smaller circuits is satisfiable. If the black-box says the circuit is not satisfiable, then none of the smaller circuits is satisfiable.

**Exercise 29.11.**

(a) We give two different types of AND circuits, illustrating the constructions for \( \ell = 8 \).

In both circuits, the number of AND-gates used is \( \ell - 1 \). The number of gates used is called the **size** of the circuit, so the sizes of our circuits are \( \ell - 1 \). A circuit is a directed graph. The **depth** of a circuit is the length of the longest path. The main difference between our two circuits is the depth. On the left, the depth is \( \ell - 1 \), linear in \( \ell \). You should convince yourself that the circuit on the right has depth \( \lceil \log_2 \ell \rceil \), logarithmic in \( \ell \).

Processor-chips are implemented using gates, starting with building blocks like multi-input-AND, adders, multipliers, sorters, etc. Designing circuits to accomplish tasks while minimizing size and depth are important considerations for efficient chip-design. Circuit complexity theorists study the minimum size and depth requirements for certain basic operations which are primitives of more complex operations. We know quite impressive designs for many basic operations, but surprisingly little about the best one can do, even for basic operations.

(b) We begin with the primitive task of moving a 1 all the way to the left. And, to simplify even further, consider just \( n = 2 \) (two input bits). On the left is a simple circuit for 2 input bits. You can verify the input-output table on the right to confirm that this circuit shifts the 1 to the left. This circuit also sorts two input-bits. Let us denote this circuit as the bubble-left (BL) "gate". The 1 in the input bubbles "up" to the left. We now combine BL-gates to build a circuit that shifts a 1 (if there is a 1) all the way to the left of an \( n \)-bit string. We call this the shift-L circuit. Suppose we have a shift-L circuit for \( n - 1 \) bits. For the \( n \) bits \( x_1, \ldots, x_n \), apply the shift-L circuit to \( x_2, \ldots, x_n \). Now, if there is a 1 in \( x_2, \ldots, x_n \), it will be on the left, at position \( x_2 \). Applying the BL-gate to \( x_1 \) and the output at the \( x_2 \) position will move the 1 at the \( x_2 \)-position to the left. Here is an illustration for a 4-bit input.
In the second step, we recursively applied the construction to shift-l(3). Observe that shift-l(2) is just a bl-gate, and so shift-l(4) is just a cascade from right to left of 3 bl-gates. In general, shift-l(n) is a cascade from right to left of n − 1 bl-gates. Since a bl-gate consists of two ∧/∨ gates, shift-l(n) uses 2n − 2 gates. The proof is by induction. The base case is shift-l(2) which is just a bl-gate that uses 2 ∧/∨ gates. For the induction, assume shift-l(n) uses 2n − 2 gates; by our construction, shift-l(n + 1) adds one bl-gate to shift-l(n), adding 2 ∧/∨ gates, resulting in a total of 2n gates. The two important properties of shift-l(n) that we need are:

Lemma 30.9. Let (y₁, ..., yₙ) = shift-l(n)(x₁, ..., xₙ). Then,

(i) (y₁, ..., yₙ) and (x₁, ..., xₙ) have the same number of 1’s;
(ii) if (x₁, ..., xₙ) contains a 1, then y₁ = 1.

In words, the output of shift-l(n) is a permutation of its input with at least one 1 having “bubbled-up” all the way to the left. We encourage you to tinker and confirm by hand that shift-l(4) applied to 0111 gives 1011. The proof of the lemma is by induction.

Proof. The base case is shift-l(2) for which the claim is true by inspecting the input-output table for the bl-gate. Suppose the claim holds for shift-l(n) and consider shift-l(n + 1) applied to (x₁, x₂, ..., xₙ₊₁) in two steps. First apply shift-l(n) to (x₂, ..., xₙ₊₁) to get (y₂, y₃, ..., yₙ₊₁). Now apply the bl-gate to (x₁, y₂) to get (y₁, y₂). By the induction hypothesis, both operations preserve the bits, so the result is a permutation. We now prove the contrapositive of property (ii). If y₁ = 0, then, from the input-output table of the bl-gate, the only possibility is x₁ = y₁′ = 0. If y₁′ = 0, then by the induction hypothesis there were no 1’s in (x₂, ..., xₙ₊₁), for if there were 1’s, the application of shift-l(n) would make y₁′ = 1. Thus, if y₁ = 0, then x₁ = x₂ = · · · = xₙ₊₁ = 0, that is there are no 1’s in the input (the contrapositive of property (ii)). Therefore, properties (i) and (ii) hold for shift-l(n + 1), and the lemma follows by induction.

We now use shift-l(n) to construct the sorter. The idea is simple. Repeatedly shift a 1 to the left. Eventually, all the 1’s will be on the left. This idea is illustrated on the right for 4 inputs. For an n-bit input, the number of gates in the sorter is

\[ 2 + 4 + \cdots + 2(n - 1) = 2 \sum_{i=1}^{n-1} i = n(n - 1) = \Theta(n^2). \]

It is helpful to see how the sorter works on input 0111.

[Diagram showing the sorter construction]

This example also illustrates why we need all the layers of shift-l, because none of the intermediate outputs are sorted. Observe that in each application of shift-l, a 1 “bubbles-up” to the left. You may have seen a sorting algorithm called bubble-sort in your earlier programming courses. We have created a circuit version of bubble-sort.

We now prove that our sorter works.

Theorem 30.10. Our construction produces a sorter using \( \Theta(n^2) \) ∧/∨ gates.
Proof. The proof is by induction, based on the observation that our sort($n$) circuit is composed of a shift-L($n$) circuit followed by a sort($n-1$) circuit applied to the last $n-1$ outputs of the shift-L($n$) circuit, as illustrated for $n = 4$ on the right. The shift-L($n$) circuit moves a 1 to the left, which becomes the output $y_1$. By the induction hypothesis, we assume that sort($n-1$) works, which means that its output is 1’s followed by 0’s, which become the outputs $y_2$, ..., $y_n$. Therefore the full output $y_1, \ldots, y_n$ is 1’s followed by zeros, which is sorted as required.

Pop Quiz 29.12.
(a) The first three bits after sorting are 1, so the and after the sorter outputs 1. It now suffices to show that every input to the and in the clique-verifier is 1: $C_{1,2} = C_{1,3} = C_{1,4} = C_{1,5} = 1$ because $x_1 = 0$; $C_{2,3} = 1$ because $e_{2,3} = 1$; $C_{2,4} = 1$ because $e_{2,4} = 1$; $C_{2,5} = 1$ because $x_5 = 0$; $C_{3,4} = 1$ because $e_{3,4} = 1$; $C_{3,5} = C_{4,5} = 1$ because $x_5 = 0$.

The output being 1 means that 01110 is a clique of size at least 3 in the graph 1010110101.

(b) Each circuit $C_{i,j}$ uses 2 gates, and there are $\frac{1}{2}n(n-1)$ of those for a total of $n(n-1)$ gates, where $n$ is the number of vertices. (we usually exclude not-gates from the size of a circuit.

If you counted not-gates, it is not a big deal.) The and in the clique-verifier therefore needs $\frac{1}{2}n(n-1) - 1$ gates, so the size of our clique verifier is $\frac{1}{2}n(n-1) - 1$. The sorter uses $n(n-1)$ gates and the and after the sorter uses $k - 1$ gates. So the total number of gates used is $\frac{3}{2}n(n-1) + k - 2 \in \Theta(n^2)$.

The input size is in $O(n^2)$, so the verifier size is linear in the input size. For $n = 5$ and $k = 3$, our formula evaluates to 51 gates. (If you counted not-gates, you should get 71 gates.)

Exercise 29.13. The construction is identical to our circuit that verified clique. The only difference is in how you construct the primitive circuit $C_{i,j}$ which ensured the “clique-condition”.

Here, we need to ensure the “independent set-condition”. For the clique-condition, the circuit $C_{i,j}$ verified that if vertices $i, j$ are in the clique, then edge $e_{i,j}$ is in the graph. For the independent set-condition, it is the opposite: if vertices $i, j$ are in the independent set, then edge $e_{i,j}$ is not in the graph. That is, for the indset problem, $C_{i,j}$ computes $\overline{e_{i,j}} \lor \overline{x_i} \lor \overline{x_j}$ (as opposed to $e_{i,j} \lor \overline{x_i} \lor \overline{x_j}$ for the clique problem). Equivalently, we want $C_{i,j}$ to compute $e_{i,j} \land x_i \land x_j$, and we give the circuit which implements this independent set-condition on the right. The rest of the circuit, including the size-verifier, is identical to that for the clique problem.

Pop Quiz 29.14. We need to prove, for every $L \in \text{NP}$, if $L^\prime$ is polynomially solvable, then $L$ is polynomially solvable.

We use a direct proof. Suppose $L^\prime$ is polynomially solvable and consider any $L \in \text{NP}$. Since (29.2) is $\tau$, it means circuitSAT is polynomially solvable. Since circuitSAT is NP-complete (the claim in (29.1) is $\tau$), it means $L$ is polynomially solvable, as was to be shown.

Pop Quiz 29.15.
(a) There is nothing to prove because 1 means $\tau$ and 0 means $F$.

(b) Suppose $u = v$, so there are two cases: $u = v = 0$ and $u = v = 1$. In both cases, $(u \lor \pi)$ and $(\pi \lor v)$ are $\tau$, therefore $u = v \rightarrow (u \lor \pi) \land (\pi \lor v)$.

Now suppose $(u \lor \pi) \land (\pi \lor v)$ is $\tau$. There are two cases. $u = 1$ and $u = 0$. If $u = 1$, then for $(\pi \lor v)$ to be $\tau$, $v = 1$ so $u = v$. If $u = 0$, then for $(u \lor \pi)$ to be $\tau$, $v = 0$ so $u = v$. In both cases $v = u$, therefore $(u \lor \pi) \land (\pi \lor v) \rightarrow u = v$.

You may wonder how we got this equivalent logical expression to $u = v$. So, we take this opportunity to introduce two fundamental representations of Boolean functions in computer science: the disjunctive normal form (DNF) which is an or of ands and the conjunctive normal form (CNF) which is an and of ors.

Both representations are based on the truth-table for $u = v$ shown on the right.

For $u = v$ to be $\tau$, either $u = 0$ and $v = 0$ or $u = 1$ and $v = 1$. That is, $u = v$ is equivalent to $(\pi \land \pi) \lor (u \land v)$ (DNF)
For \( u = v \) to be \( T \), either \( u = 0 \) and \( v = 1 \) or \( u = 1 \) and \( v = 0 \). That is,

\[
u = v \iff (\neg v \lor v) \lor (u \land v)
\]

This means \( u = v \) is equivalent to \( (\neg v \lor v) \lor (u \land v) \). We now use De Morgan’s laws for negation, namely \( A \lor B \equiv \neg A \land \neg B \) and \( A \land B \equiv \neg A \lor \neg B \) to get

\[
u = v \iff (u \lor v) \land (\neg u \land v) \quad \text{(CNF)}
\]

(c) Suppose \( u = v \lor w \). There are four cases corresponding to \((v, w)\) being \((0, 0), (0, 1), (1, 0), (1, 1)\).
In each case \((u \lor v) \land (u \lor w) \land (v \lor w)\) is \( T \).
Now suppose \((u \lor v) \land (u \lor w) \land (v \lor w)\) is \( T \). There are two cases, \( u = 1 \) or \( u = 0 \). If \( u = 1 \), then for \((\neg u \land v) \land (u \lor w)\) to be \( T \), \((v \lor w)\) must be \( T \), that is \( u = v \lor w \). If \( u = 0 \), then for \((u \lor v) \land (v \lor w)\) to be \( T \), \( v = 0 \) and \((u \lor w)\), \( w = 0 \). So \( v \lor w = 0 \), i.e. \( u = v \lor w \).

It is a useful exercise to recover this logical expression for \( u = v \lor w \) from its truth-table using the CNF construction.

(d) This follows from (c) because \( u = v \land w \) if and only if \( v = \neg v \land w \equiv v \lor w \). So, we get the expression in (c) with \( u, v, w \) replaced by \( \pi, \pi, \pi \).

**Pop Quiz 29.16.**

- clause 1
  
  \[
y \equiv \pi \quad \pi \quad \pi \quad \pi \rightarrow (x, y, z) = (T, T, T);
\]

- clause 2
  
  \[
y \equiv \pi \quad \pi \quad \pi \quad \pi \rightarrow (x, y, z) = (F, T, F);
\]

- clause 3
  
  \[
z \equiv \pi \quad \pi \quad \pi \quad \pi \rightarrow (x, y, z) = (F, F, T);
\]

- clause 4
  
  \[
z \equiv \pi \quad \pi \quad \pi \quad \pi \rightarrow (x, y, z) = (F, F, F);
\]

**Exercise 29.17.** Only problems in \( \text{NP} \) can be \( \text{NP-complete} \). So when trying to show that a problem is \( \text{NP-complete} \), you should always verify first that the problem is in \( \text{NP} \).

(a) We know that \text{CLIQUE} is in \( \text{NP} \). For the \( \text{NP-complete} \) problem \( \mathcal{L}^* \), we choose \text{IndSet}. We show that \text{IndSet} is polynomially reducible to \text{CLIQUE}.

Recall the complement graph \( G \) of a graph \( \tilde{G} \). The complement graph is obtained from the original graph by removing all existing edges and adding all other edges:

\[
(u, v) \text{ is an edge in } \tilde{G} \iff (u, v) \text{ is not an edge in } G.
\]

An independent set in the graph is a clique in the complement graph and vice versa. Therefore, \( G \) has an independent set of size \( k \) if and only if the complement graph has a clique of size \( k \).

If we have a black-box which polynomially solves \text{CLIQUE}, we can solve \text{IndSet} by using the black-box on the complement graph. Therefore,

**Theorem 30.11 (CLIQUE is \text{NP-complete}).** If \text{CLIQUE} is polynomially solvable, then \text{IndSet} is polynomially solvable.

(b) First, let’s check that \text{VERTEXCOVER} is in \( \text{NP} \). By checking each edge, one can verify in polynomial time whether a set is a vertex cover (every edge should have at least one end in the vertex cover), and all that remains is to check the size of the vertex cover. Therefore a (\text{YES})-instance of \text{VERTEXCOVER} can be polynomially verified given the evidence, which is the vertex cover itself.

For the \text{NP-complete} problem \( \mathcal{L}^* \), we again choose \text{IndSet}. We show that \text{IndSet} is polynomially reducible to \text{VERTEXCOVER}. Assume a graph has \( n \) vertices \( V = \{v_1, \ldots, v_n\} \), and let \( S \subseteq V \) be a subset of the vertices. The complement of \( S \) is \( \overline{S} = V - S \).

**Lemma 30.12.** \( S \) is an independent set of size \( k \) if and only if \( \overline{S} \) is a vertex cover of size \( n - k \).

**Proof.** Suppose \( S \) is an independent set of size \( k \). Consider any edge \( e \). If both endpoints of \( e \) are in \( S \) then \( S \) is not an independent set (there cannot be an edge between any two vertices in \( S \)). Therefore at least one endpoint of \( e \) is in \( \overline{S} \). Since \( e \) is arbitrary, every edge has at least one endpoint in \( \overline{S} \), and so \( \overline{S} \) is a vertex cover of size \( n - k \).

Suppose \( \overline{S} \) is a vertex cover of size \( n - k \). Now consider any pair of vertices in \( S \). If there is an edge between these two vertices, then that edge does not have an endpoint in \( \overline{S} \), contradicting \( \overline{S} \) being a vertex cover. Therefore no pair of vertices in \( S \) is adjacent and so \( S \) is an independent set of size \( k \).

Suppose we have a black-box to solve \text{VERTEXCOVER}. We solve \text{IndSet} with size \( k \) by running the black-box on the same graph with size \( n - k \) (seeking a vertex cover of size at most \( n - k \)).

Lemma 30.12 assures us that this algorithm works.
Theorem 30.13 (VertexCover is NP-complete). If VertexCover is polynomially solvable, then IndSet is polynomially solvable.

Exercise 29.18.

(a) This is an NP-completeness reduction from a general problem (Clique) to a restriction of the problem to special cases (in BigClique, k must be large). We need to show that if we have a polynomial black-box that solves BigClique we can polynomially solve Clique.

Consider any instance \( G, k \) of Clique and suppose \( G \) has \( n \) vertices \( v_1, \ldots, v_n \). We cannot use our black-box to solve this problem because \( k \) may be too small. Our solution is to convert this clique problem to another equivalent one with a large \( k \). Construct a new graph \( G' \) by adding \( n \) new vertices \( w_1, \ldots, w_n \) to \( G \). Also add edges from every \( w \)-vertex to every other vertex in \( G' \). Clearly, it takes polynomial time to construct \( G' \) from \( G \).

If you have any \( k \)-clique in \( G \), then those vertices together with all the \( w \)-vertices form a \((k + n)\)-clique in \( G' \). Further, every \((k + n)\)-clique in \( G' \) contains at least \( k \) \( v \)-vertices which are all connected to each other and so contain a \( k \)-clique in \( G \). Therefore,

**Lemma 30.14.** \( G \) contains a \( k \)-clique if and only if \( G' \) contains a \((k + n)\)-clique.

Here is the algorithm to solve Clique. First construct \( G' \). Now run the black-box on \( G' \) with \( k' = k + n \) (which is larger than half the vertices in \( G' \)). Output the answer [YES/NO] from the black-box. The Lemma ensures that this answer is correct.

(b) Suppose you have a black-box which polynomially solves FreqItems.\(^1\) We show how to use the black-box to polynomially solve Clique. It will follow that FreqItems is NP-complete.

Given an input \( G = (V, E), k \) to Clique, we construct an input to FreqItems, a binary matrix \( A \), popularity \( n \) and basket-size \( \ell \). We run our black-box on this input of FreqItems, and the answer will tell us whether \( G \) has a \( k \)-clique.

The customers will be the vertices in \( G \). The items will be the edges in \( G \). Each of the vertices (customers) buys an edge (item) whenever the customer is not an end point of the edge. Here is an example. The input to Clique is on the left; \( A \) is on the right.

\[
\begin{align*}
G; k &= 4 \\
A; n &= 2, \ell = 6
\end{align*}
\]

We highlighted a solution to FreqItems with a basket of size \( \ell = 6 \) bought by \( n = 2 \) customers. The other 4 customers (nodes) must form a 4-clique in \( G \) because the 6 edges in our basket (bought by two customers) have endpoints among the 4 customers who did not buy the edges. The maximum number of edges in a group of 4 vertices is 6, which can only be so if every edge is present. That is, those 4 vertices are a 4-clique.

For a general instance of Clique, we construct an instance \( A \) of FreqItems as we described. Now run our black-box on \( A \) with \( n = |V| - k \) and \( \ell = \frac{1}{2}k(k - 1) \) to get the answer to the instance of Clique. The next lemma proves the answer is correct. Further, since the black-box is polynomial, the entire solution to Clique is polynomial.

**Lemma 30.15.** There is a \( k \)-clique in \( G = (V, E) \) if and only if the instance \( A \) of FreqItems that we constructed has a basket of size \( \frac{1}{2}k(k - 1) \) which is bought by \( |V| - k \) customers.

**Proof.** Suppose there is a \( k \)-clique in \( G = (V, E) \). In \( A \), the \( \frac{1}{2}k(k - 1) \) edges in the \( k \)-clique is a basket of edges which is bought by each of the \( |V| - k \) vertices (customers) not in the clique.

Suppose that in \( A \) there is a basket of size \( \frac{1}{2}k(k - 1) \) which is bought by \( |V| - k \) customers. Those \( \frac{1}{2}k(k - 1) \) edges in the basket only connect to the \( k \) customers who did not by the

---

\(^1\)FreqItems is more general than BalancedBipartiteClique which is the NP-complete problem GT24 in the Garey & Johnson book *Computers and Intractability*. A more general problem cannot be easier.
basket. Therefore, those $k$ customers are a subgraph with $\frac{1}{2}k(k - 1)$ edges, which can only be the case if every edge is present, and they form a $k$-clique in $G$. ■

Let us summarize the general methodology for proving a problem is NP-complete.

To show $L$ is NP-complete using the NP-complete problem $L^*$
1: Start with a general instance $I^*$ of $L^*$.
2: Construct an instance $I$ of $L$ from $I^*$, in polynomial time.
3: Show that $I^* \in L^*$ if and only if $I \in L$. 

550