a) Show that this algorithm uses $O(n^3)$ comparisons to compute the matrix $\mathbf{M}$.

b) Show that this algorithm uses $\Omega(n^3)$ comparisons to compute the matrix $\mathbf{M}$. Using this fact and part (a), conclude that the algorithms uses $\Theta(n^3)$ comparisons. [*Hint:* Only consider the cases where $i \leq n/4$ and $j \geq 3n/4$ in the two outer loops in the algorithm.]

13. The conventional algorithm for evaluating a polynomial $a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$ at $x = c$ can be expressed in pseudocode by

procedure *polynomial*($c, a_0, a_1, \ldots, a_n$: real numbers)
  power := 1
  $y := a_0$
  for i := 1 **to** n
    power := *power* $*$ c
    $y := y + a_i * power$
  return y $\{y = a_n c^n + a_{n-1}c^{n-1} + \cdots + a_1 c + a_0\}$

where the final value of y is the value of the polynomial at $x = c$.

a) Evaluate $3x^2 + x + 1$ at $x = 2$ by working through each step of the algorithm showing the values assigned at each assignment step.

b) Exactly how many multiplications and additions are used to evaluate a polynomial of degree n at $x = c$? (Do not count additions used to increment the loop variable.)

14. There is a more efficient algorithm (in terms of the number of multiplications and additions used) for evaluating polynomials than the conventional algorithm described in the previous exercise. It is called **Horner's method**. This pseudocode shows how to use this method to find the value of $a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$ at $x = c$.

procedure *Horner*($c, a_0, a_1, a_2, \ldots, a_n$: real numbers)
  $y := a_n$
  for i := 1 **to** n
    $y := y * c + a_{n-i}$
  return y $\{y = a_n c^n + a_{n-1}c^{n-1} + \cdots + a_1 c + a_0\}$

a) Evaluate $3x^2 + x + 1$ at $x = 2$ by working through each step of the algorithm showing the values assigned at each assignment step.

b) Exactly how many multiplications and additions are used by this algorithm to evaluate a polynomial of degree n at $x = c$? (Do not count additions used to increment the loop variable.)

15. What is the largest n for which one can solve within one second a problem using an algorithm that requires $f(n)$ bit operations, where each bit operation is carried out in 10^{-9} seconds, with these functions $f(n)$?

a) $\log n$ **b)** n **c)** $n \log n$
d) n^2 **e)** 2^n **f)** $n!$

16. What is the largest n for which one can solve within a day using an algorithm that requires $f(n)$ bit operations, where each bit operation is carried out in 10^{-11} seconds, with these functions $f(n)$?

a) $\log n$ **b)** $1000n$ **c)** n^2
d) $1000n^2$ **e)** n^3 **f)** 2^n
g) 2^{2n} **h)** 2^{2^n}

17. What is the largest n for which one can solve within a minute using an algorithm that requires $f(n)$ bit operations, where each bit operation is carried out in 10^{-12} seconds, with these functions $f(n)$?

a) $\log \log n$ **b)** $\log n$ **c)** $(\log n)^2$
d) $1000000n$ **e)** n^2 **f)** 2^n
g) 2^{n^2}

18. How much time does an algorithm take to solve a problem of size n if this algorithm uses $2n^2 + 2^n$ operations, each requiring 10^{-9} seconds, with these values of n?

a) 10 **b)** 20 **c)** 50 **d)** 100

19. How much time does an algorithm using 2^{50} operations need if each operation takes these amounts of time?

a) 10^{-6} s **b)** 10^{-9} s **c)** 10^{-12} s

20. What is the effect in the time required to solve a problem when you double the size of the input from n to $2n$, assuming that the number of milliseconds the algorithm uses to solve the problem with input size n is each of these function? [Express your answer in the simplest form possible, either as a ratio or a difference. Your answer may be a function of n or a constant.]

a) $\log \log n$ **b)** $\log n$ **c)** $100n$
d) $n \log n$ **e)** n^2 **f)** n^3
g) 2^n

21. What is the effect in the time required to solve a problem when you increase the size of the input from n to $n + 1$, assuming that the number of milliseconds the algorithm uses to solve the problem with input size n is each of these function? [Express your answer in the simplest form possible, either as a ratio or a difference. Your answer may be a function of n or a constant.]

a) $\log n$ **b)** $100n$ **c)** n^2
d) n^3 **e)** 2^n **f)** 2^{n^2}
g) $n!$

22. Determine the least number of comparisons, or best-case performance,

a) required to find the maximum of a sequence of n integers, using Algorithm 1 of Section 3.1.

b) used to locate an element in a list of n terms with a linear search.

c) used to locate an element in a list of n terms using a binary search.

23. Analyze the average-case performance of the linear search algorithm, if exactly half the time the element x is not in the list and if x is in the list it is equally likely to be in any position.

24. An algorithm is called **optimal** for the solution of a problem with respect to a specified operation if there is no algorithm for solving this problem using fewer operations.

a) Show that Algorithm 1 in Section 3.1 is an optimal algorithm with respect to the number of comparisons of integers. [*Note:* Comparisons used for bookkeeping in the loop are not of concern here.]

b) Is the linear search algorithm optimal with respect to the number of comparisons of integers (not including comparisons used for bookkeeping in the loop)?

25. Describe the worst-case time complexity, measured in terms of comparisons, of the ternary search algorithm described in Exercise 27 of Section 3.1.

26. Describe the worst-case time complexity, measured in terms of comparisons, of the search algorithm described in Exercise 28 of Section 3.1.

27. Analyze the worst-case time complexity of the algorithm you devised in Exercise 29 of Section 3.1 for locating a mode in a list of nondecreasing integers.

28. Analyze the worst-case time complexity of the algorithm you devised in Exercise 30 of Section 3.1 for locating all modes in a list of nondecreasing integers.

29. Analyze the worst-case time complexity of the algorithm you devised in Exercise 31 of Section 3.1 for finding the first term of a sequence of integers equal to some previous term.

30. Analyze the worst-case time complexity of the algorithm you devised in Exercise 32 of Section 3.1 for finding all terms of a sequence that are greater than the sum of all previous terms.

31. Analyze the worst-case time complexity of the algorithm you devised in Exercise 33 of Section 3.1 for finding the first term of a sequence less than the immediately preceding term.

32. Determine the worst-case complexity in terms of comparisons of the algorithm from Exercise 5 in Section 3.1 for determining all values that occur more than once in a sorted list of integers.

33. Determine the worst-case complexity in terms of comparisons of the algorithm from Exercise 9 in Section 3.1 for determining whether a string of n characters is a palindrome.

34. How many comparisons does the selection sort (see preamble to Exercise 41 in Section 3.1) use to sort n items? Use your answer to give a big-O estimate of the complexity of the selection sort in terms of number of comparisons for the selection sort.

35. Find a big-O estimate for the worst-case complexity in terms of number of comparisons used and the number of terms swapped by the binary insertion sort described in the preamble to Exercise 47 in Section 3.1.

36. Show that the greedy algorithm for making change for n cents using quarters, dimes, nickels, and pennies has $O(n)$ complexity measured in terms of comparisons needed.

Exercises 37 and 38 deal with the problem of scheduling the most talks possible given the start and end times of n talks.

37. Find the complexity of a brute-force algorithm for scheduling the talks by examining all possible subsets of the talks. [*Hint:* Use the fact that a set with n elements has 2^n subsets.]

38. Find the complexity of the greedy algorithm for scheduling the most talks by adding at each step the talk with the earliest end time compatible with those already scheduled (Algorithm 7 in Section 3.1). Assume that the talks are not already sorted by earliest end time and assume that the worst-case time complexity of sorting is $O(n \log n)$.

39. Describe how the number of comparisons used in the worst case changes when these algorithms are used to search for an element of a list when the size of the list doubles from n to $2n$, where n is a positive integer.

a) linear search **b)** binary search

40. Describe how the number of comparisons used in the worst case changes when the size of the list to be sorted doubles from n to $2n$, where n is a positive integer when these sorting algorithms are used.

a) bubble sort **b)** insertion sort

c) selection sort (described in the preamble to Exercise 41 in Section 3.1)

d) binary insertion sort (described in the preamble to Exercise 47 in Section 3.1)

An $n \times n$ matrix is called **upper triangular** if $a_{ij} = 0$ whenever $i > j$.

41. From the definition of the matrix product, describe an algorithm in English for computing the product of two upper triangular matrices that ignores those products in the computation that are automatically equal to zero.

42. Give a pseudocode description of the algorithm in Exercise 41 for multiplying two upper triangular matrices.

43. How many multiplications of entries are used by the algorithm found in Exercise 41 for multiplying two $n \times n$ upper triangular matrices?

In Exercises 44–45 assume that the number of multiplications of entries used to multiply a $p \times q$ matrix and a $q \times r$ matrix is pqr.

44. What is the best order to form the product **ABC** if **A**, **B**, and **C** are matrices with dimensions 3×9, 9×4, and 4×2, respectively?

45. What is the best order to form the product **ABCD** if **A**, **B**, **C**, and **D** are matrices with dimensions 30×10, 10×40, 40×50, and 50×30, respectively?.

***46.** In this exercise we deal with the problem of **string matching**.

a) Explain how to use a brute-force algorithm to find the first occurrence of a given string of m characters, called the **target**, in a string of n characters, where $m \leq n$, called the **text**. [*Hint:* Think in terms of finding a match for the first character of the target and checking successive characters for a match, and if they do not all match, moving the start location one character to the right.]

b) Express your algorithm in pseudocode.

c) Give a big-O estimate for the worst-case time complexity of the brute-force algorithm you described.

Key Terms and Results

TERMS

algorithm: a finite sequence of precise instructions for performing a computation or solving a problem

searching algorithm: the problem of locating an element in a list

linear search algorithm: a procedure for searching a list element by element

binary search algorithm: a procedure for searching an ordered list by successively splitting the list in half

sorting: the reordering of the elements of a list into prescribed order

$f(x)$ is $O(g(x))$: the fact that $|f(x)| \leq C|g(x)|$ for all $x > k$ for some constants C and k

witness to the relationship $f(x)$ is $O(g(x))$: a pair C and k such that $|f(x)| \leq C|g(x)|$ whenever $x > k$

$f(x)$ is $\Omega(g(x))$: the fact that $|f(x)| \geq C|g(x)|$ for all $x > k$ for some positive constants C and k

$f(x)$ is $\Theta(g(x))$: the fact that $f(x)$ is both $O(g(x))$ and $\Omega(g(x))$

time complexity: the amount of time required for an algorithm to solve a problem

space complexity: the amount of space in computer memory required for an algorithm to solve a problem

worst-case time complexity: the greatest amount of time required for an algorithm to solve a problem of a given size

average-case time complexity: the average amount of time required for an algorithm to solve a problem of a given size

algorithmic paradigm: a general approach for constructing algorithms based on a particular concept

brute force: the algorithmic paradigm based on constructing algorithms for solving problems in a naive manner from the statement of the problem and definitions

greedy algorithm: an algorithm that makes the best choice at each step according to some specified condition

tractable problem: a problem for which there is a worst-case polynomial-time algorithm that solves it

intractable problem: a problem for which no worst-case polynomial-time algorithm exists for solving it

solvable problem: a problem that can be solved by an algorithm

unsolvable problem: a problem that cannot be solved by an algorithm

RESULTS

linear and binary search algorithms: (given in Section 3.1)

bubble sort: a sorting that uses passes where successive items are interchanged if they in the wrong order

insertion sort: a sorting that at the jth step inserts the jth element into the correct position in in the list, when the first $j-1$ elements of the list are already sorted

The linear search has $O(n)$ worst case time complexity.

The binary search has $O(\log n)$ worst case time complexity.

The bubble and insertion sorts have $O(n^2)$ worst case time complexity.

$\log n!$ is $O(n \log n)$.

If $f_1(x)$ is $O(g_1(x))$ and $f_2(x)$ is $O(g_2(x))$, then $(f_1 + f_2)(x)$ is $O(\max(g_1(x), g_2(x)))$ and $(f_1 f_2)(x)$ is $O((g_1 g_2(x))$.

If $a_0, a_1, \ldots, a_n$ are real numbers with $a_n \neq 0$, then $a_n x^n + a_{n-1}x^{n-1} + \cdots + a_1 x + a_0$ is $\Theta(x^n)$, and hence $O(n)$ and $\Omega(n)$.

Review Questions

1. a) Define the term *algorithm*.

b) What are the different ways to describe algorithms?

c) What is the difference between an algorithm for solving a problem and a computer program that solves this problem?

2. a) Describe, using English, an algorithm for finding the largest integer in a list of n integers.

b) Express this algorithm in pseudocode.

c) How many comparisons does the algorithm use?

3. a) State the definition of the fact that $f(n)$ is $O(g(n))$, where $f(n)$ and $g(n)$ are functions from the set of positive integers to the set of real numbers.

b) Use the definition of the fact that $f(n)$ is $O(g(n))$ directly to prove or disprove that $n^2 + 18n + 107$ is $O(n^3)$.

c) Use the definition of the fact that $f(n)$ is $O(g(n))$ directly to prove or disprove that n^3 is $O(n^2 + 18n + 107)$.

4. List these functions so that each function is big-O of the next function in the list: $(\log n)^3$, $n^3/1000000$, $\sqrt{n}$, $100n + 101$, 3^n, $n!$, $2^n n^2$.

5. a) How can you produce a big-O estimate for a function that is the sum of different terms where each term is the product of several functions?

b) Give a big-O estimate for the function $f(n) = (n! + 1)(2^n + 1) + (n^{n-2} + 8n^{n-3})(n^3 + 2^n)$. For the function g in your estimate $f(x)$ is $O(g(x))$ use a simple function of smallest possible order.

6. a) Define what the worst-case time complexity, average-case time complexity, and best-case time complexity (in terms of comparisons) mean for an algorithm that finds the smallest integer in a list of n integers.

b) What are the worst-case, average-case, and best-case time complexities, in terms of comparisons, of the algorithm that finds the smallest integer in a list of n integers by comparing each of the integers with the smallest integer found so far?

7. a) Describe the linear search and binary search algorithm for finding an integer in a list of integers in increasing order.
b) Compare the worst-case time complexities of these two algorithms.
c) Is one of these algorithms always faster than the other (measured in terms of comparisons)?

8. a) Describe the bubble sort algorithm.
b) Use the bubble sort algorithm to sort the list 5, 2, 4, 1, 3.
c) Give a big-O estimate for the number of comparisons used by the bubble sort.

9. a) Describe the insertion sort algorithm.
b) Use the insertion sort algorithm to sort the list 2, 5, 1, 4, 3.
c) Give a big-O estimate for the number of comparisons used by the insertion sort.

10. a) Explain the concept of a greedy algorithm.
b) Provide an example of a greedy algorithm that produces an optimal solution and explain why it produces an optimal solution.
c) Provide an example of a greedy algorithm that does not always produce an optimal solution and explain why it fails to do so.

11. Define what it means for a problem to be tractable and what it means for a problem to be solvable.

Supplementary Exercises

1. a) Describe an algorithm for locating the last occurrence of the largest number in a list of integers.
b) Estimate the number of comparisons used.

2. a) Describe an algorithm for finding the first and second largest elements in a list of integers.
b) Estimate the number of comparisons used.

3. a) Give an algorithm to determine whether a bit string contains a pair of consecutive zeros.
b) How many comparisons does the algorithm use?

4. a) Suppose that a list contains integers that are in order of largest to smallest and an integer can appear repeatedly in this list. Devise an algorithm that locates all occurrences of an integer x in the list.
b) Estimate the number of comparisons used.

5. a) Adapt Algorithm 1 in Section 3.1 to find the maximum and the minimum of a sequence of n elements by employing a temporary maximum and a temporary minimum that is updated as each successive element is examined.
b) Describe the algorithm from part (a) in pseudocode.
c) How many comparisons of elements in the sequence are carried out by this algorithm? (Do not count comparisons used to determine whether the end of the sequence has been reached.)

6. a) Describe in detail (and in English) the steps of an algorithm that finds the maximum and minimum of a sequence of n elements by examining pairs of successive elements, keeping track of a temporary maximum and a temporary minimum. If n is odd, both the temporary maximum and temporary minimum should initially equal the first term, and if n is even, the temporary minimum and temporary maximum should be found by comparing the initial two elements. The temporary maximum and temporary minimum should be updated by comparing them with the maximum and minimum of the pair of elements being examined.
b) Express the algorithm described in part (a) in pseudocode.
c) How many comparisons of elements of the sequence are carried out by this algorithm? (Do not count comparisons used to determine whether the end of the sequence has been reached.) How does this compare to the number of comparisons used by the algorithm in Exercise 5?

***7.** Show that the worst-case complexity in terms of comparisons of an algorithm that finds the maximum and minimum of n elements is at least $\lceil 3n/2 \rceil - 2$.

8. Devise an efficient algorithm for finding the second largest element in a sequence of n elements and determine the worst-case complexity of your algorithm.

9. Devise an algorithm that finds all equal pairs of sums of two terms of a sequence of n numbers, and determine the worst-case complexity of your algorithm.

10. Devise an algorithm that finds the closest pair of integers in a sequence of n integers, and determine the worst-case complexity of your algorithm. [*Hint:* Sort the sequence. Use the fact that sorting can be done with worst-case time complexity $O(n \log n)$.]

The **shaker sort** (or **bidirectional bubble sort**) successively compares pairs of adjacent elements, exchanging them if they are out of order, and alternately passing through the list from the beginning to the end and then from the end to the beginning until no exchanges are needed.

11. Show the steps used by the shaker sort to sort the list 3, 5, 1, 4, 6, 2.

12. Express the shaker sort in pseudocode.

13. Show that the shaker sort has $O(n^2)$ complexity measured in terms of the number of comparisons it uses.

14. Explain why the shaker sort is efficient for sorting lists that are already in close to the correct order.

15. Show that $(n \log n + n^2)^3$ is $O(n^6)$.

16. Show that $8x^3 + 12x + 100 \log x$ is $O(x^3)$.

17. Give a big-O estimate for $(x^2 + x(\log x)^3) \cdot (2^x + x^3)$.

18. Find a big-O estimate for $\sum_{j=1}^{n} j(j+1)$.

***19.** Show that $n!$ is not $O(2^n)$.

***20.** Show that n^n is not $O(n!)$.

21. Find all pairs of functions of the same order in this list of functions: $n^2 + (\log n)^2$, $n^2 + n$, $n^2 + \log 2^n + 1$, $(n+1)^3 - (n-1)^3$, and $(n + \log n)^2$.

22. Find all pairs of functions of the same order in this list of functions $n^2 + 2^n, n^2 + 2^{100}, n^2 + 2^{2n}, n^2 + n!, n^2 + 3^n$, and $(n^2 + 1)^2$.

23. Find an integer n with $n > 2$ for which $n^{2^{100}} < 2^n$.

24. Find an integer n with $n > 2$ for which $(\log n)^{2^{100}} < \sqrt{n}$.

***25.** Arrange the functions n^n, $(\log n)^2$, $n^{1.0001}$, $(1.0001)^n$, $2^{\sqrt{\log_2 n}}$, and $n(\log n)^{1001}$ in a list so that each function is big-O of the next function. [*Hint:* To determine the relative size of some of these functions, take logarithms.]

***26.** Arrange the function 2^{100n}, 2^{n^2}, $2^{n!}$, 2^{2^n}, $n^{\log n}$, $n \log n \log \log n$, $n^{3/2}$, $n(\log n)^{3/2}$, and $n^{4/3}(\log n)^2$ in a list so that each function is big-O of the next function. [*Hint:* To determine the relative size of some of these functions, take logarithms.]

***27.** Give an example of two increasing functions $f(n)$ and $g(n)$ from the set of positive integers to the set of positive integers such that neither $f(n)$ is $O(g(n))$ nor $g(n)$ is $O(f(n))$.

28. Show that if the denominations of coins are $c^0, c^1, \ldots, c^k$, where k is a positive integer and c is a positive integer, $c > 1$, the greedy algorithm always produces change using the fewest coins possible.

29. a) Use pseudocode to specify a brute-force algorithm that determines when given as input a sequence of n positive integers whether there are two distinct terms of the sequence that have as sum a third term. The algorithm should loop through all triples of terms of the sequence, checking whether the sum of the first two terms equals the third.

b) Give a big-O estimate for the complexity of the brute-force algorithm from part (a).

30. a) Devise a more efficient algorithm for solving the problem described in Exercise 29 that first sorts the input sequence and then checks for each pair of terms whether their difference is in the sequence.

b) Give a big-O estimate for the complexity of this algorithm. Is it more efficient than the brute-force algorithm from Exercise 29?

Suppose we have s men and s women each with their preference lists for the members of the opposite gender, as described in the preamble to Exercise 60 in Section 3.1. We say that a woman w is a **valid partner** for a man m if there is some stable matching in which they are paired. Similarly, a man m is a **valid partner** for a woman w if there is some stable matching in which they are paired. A matching in which each man is assigned his valid partner ranking highest on his preference list is called **male optimal**, and a matching in which each woman is assigned her valid partner ranking lowest on her preference list is called **female pessimal**.

31. Find all valid partners for each man and each woman if there are three men m_1, m_2, and m_3 and three women w_1, w_2, w_3 with these preference rankings of the men for the women, from highest to lowest: m_1: w_3, w_1, w_2; m_2: w_3, w_2, w_1; m_3: w_2, w_3, w_1; and with these preference rankings of the women for the men, from highest to lowest: w_1: m_3, m_2, m_1; w_2: m_1, m_3, m_2; w_3: m_3, m_2, m_1.

***32.** Show that the deferred acceptance algorithm given in the preamble to Exercise 61 of Section 3.1, always produces a male optimal and female pessimal matching.

33. Define what it means for a matching to be female optimal and for a matching to be male pessimal.

***34.** Show that when woman do the proposing in the deferred acceptance algorithm, the matching produced is female optimal and male pessimal.

In Exercises 35 and 36 we consider variations on the problem of finding stable matchings of men and women described in the preamble to Exercise 61 in Section 3.1.

***35.** In this exercise we consider matching problems where there may be different numbers of men and women, so that it is impossible to match everyone with a member of the opposite gender.

a) Extend the definition of a stable matching from that given in the preamble to Exercise 60 in Section 3.1 to cover the case where there are unequal numbers of men and women. Avoid all cases where a man and a woman would prefer each other to their current situation, including those involving unmatched people. (Assume that an unmatched person prefers a match with a member of the opposite gender to remaining unmatched.)

b) Adapt the deferred acceptance algorithm to find stable matchings, using the definition of stable matchings from part (a), when there are different numbers of men and women.

c) Prove that all matchings produced by the algorithm from part (b) are stable, according to the definition from part (a).

***36.** In this exercise we consider matching problems where some man-woman pairs are not allowed.

a) Extend the definition of a stable matching to cover the situation where there are the same number of men and women, but certain pairs of men and women are forbidden. Avoid all cases where a man and a woman would prefer each other to their current situation, including those involving unmatched people.

b) Adapt the deferred acceptance algorithm to find stable matchings when there are the same number of men and women, but certain man-woman pairs are forbidden. Be sure to consider people who are unmatched at the end of the algorithm. (Assume that an unmatched person prefers a match with a member of the opposite gender who is not a forbidden partner to remaining unmatched.)

c) Prove that all matchings produced by the algorithm from (b) are stable, according to the definition in part (a).

Exercises 37–40 deal with the problem of scheduling n jobs on a single processor. To complete job j, the processor must run job j for time t_j without interruption. Each job has a deadline d_j. If we start job j at time s_j, it will be completed at time $e_j = s_j + t_j$. The **lateness** of the job measures how long it finishes after its deadline, that is, the lateness of job j is $\max(0, e_j - d_j)$. We wish to devise a greedy algorithm that minimizes the maximum lateness of a job among the n jobs.

37. Suppose we have five jobs with specified required times and deadlines: $t_1 = 25, d_1 = 50$; $t_2 = 15, d_2 = 60$; $t_3 = 20, d_3 = 60$; $t_4 = 5, d_4 = 55$; $t_5 = 10, d_5 = 75$. Find the maximum lateness of any job when the jobs are scheduled in this order (and they start at time 0): Job 3, Job 1, Job 4, Job 2, Job 5. Answer the same question for the schedule Job 5, Job 4, Job 3, Job 1, Job 2.

38. The **slackness** of a job requiring time t and with deadline d is $d - t$, the difference between its deadline and the time it requires. Find an example that shows that scheduling jobs by increasing slackness does not always yield a schedule with the smallest possible maximum lateness.

39. Find an example that shows that scheduling jobs in order of increasing time required does not always yield a schedule with the smallest possible maximum lateness.

***40.** Prove that scheduling jobs in order of increasing deadlines always produces a schedule that minimizes the maximum lateness of a job. [*Hint:* First show that for a schedule to be optimal, jobs must be scheduled with no idle time between them and so that no job is scheduled before another with an earlier deadline.]

41. Suppose that we have a knapsack with total capacity of W kg. We also have n items where item j has mass w_j. The **knapsack problem** asks for a subset of these n items with the largest possible total mass not exceeding W.

a) Devise a brute-force algorithm for solving the knapsack problem.

b) Solve the knapsack problem when the capacity of the knapsack is 18 kg and there are five items: a 5-kg sleeping bag, an 8-kg tent, a 7-kg food pack, a 4-kg container of water, and an 11-kg portable stove.

In Exercises 42–46 we will study the problem of load balancing. The input to the problem is a collection of p processors and n jobs, t_j is the time required to run job j, jobs run without interruption on a single machine until finished, and a processor can run only one job at a time. The **load** L_k of processor k is the sum over all jobs assigned to processor k of the times required to run these jobs. The **makespan** is the maximum load over all the p processors. The load balancing problem asks for an assignment of jobs to processors to minimize the makespan.

42. Suppose we have three processors and five jobs requiring times $t_1 = 3$, $t_2 = 5$, $t_3 = 4$, $t_4 = 7$, and $t_5 = 8$. Solve the load balancing problem for this input by finding the assignment of the five jobs to the three processors that minimizes the makespan.

43. Suppose that L^* is the minimum makespan when p processors are given n jobs, where t_j is the time required to run job j.

a) Show that $L^* \geq \max_{j=1,2,\ldots,n} t_j$.

b) Show that $L^* \geq \frac{1}{p} \sum_{j=1}^{n} t_j$.

44. Write out in pseudocode the greedy algorithm that goes through the jobs in order and assigns each job to the processor with the smallest load at that point in the algorithm.

45. Run the algorithm from Exercise 44 on the input given in Exercise 42.

An **approximation algorithm** for an optimization problem produces a solution guaranteed to be close to an optimal solution. More precisely, suppose that the optimization problem asks for an input S that minimizes $F(X)$ where F is some function of the input X. If an algorithm always finds an input T with $F(T) \leq cF(S)$ where c is a fixed positive real number, the algorithm is called a ***c*-approximation algorithm** for the problem.

***46.** Prove that the algorithm from Exercise 44 is a 2-approximation algorithm for the load balancing problem. [*Hint:* Use both parts of Exercise 43.]

Computer Projects

Write programs with these inputs and outputs.

1. Given a list of n integers, find the largest integer in the list.

2. Given a list of n integers, find the first and last occurrences of the largest integer in the list.

3. Given a list of n distinct integers, determine the position of an integer in the list using a linear search.

4. Given an ordered list of n distinct integers, determine the position of an integer in the list using a binary search.

5. Given a list of n integers, sort them using a bubble sort.

6. Given a list of n integers, sort them using an insertion sort.

7. Given an integer n, use the greedy algorithm to find the change for n cents using quarters, dimes, nickels, and pennies.

8. Given the starting and ending times of n talks, use the appropriate greedy algorithm to schedule the most talks possible in a single lecture hall.

9. Given an ordered list of n integers and an integer x in the list, find the number of comparisons used to determine the position of x in the list using a linear search and using a binary search.
10. Given a list of integers, determine the number of comparisons used by the bubble sort and by the insertion sort to sort this list.

Computations and Explorations

Use a computational program or programs you have written to do these exercises.

1. We know that n^b is $O(d^n)$ when b and d are positive numbers with $d \geq 2$. Give values of the constants C and k such that $n^b \leq Cd^n$ whenever $x > k$ for each of these sets of values: $b = 10, d = 2$; $b = 20, d = 3$; $b = 1000, d = 7$.
2. Compute the change for different values of n with coins of different denominations using the greedy algorithm and determine whether the smallest number of coins was used. Can you find conditions so that the greedy algorithm is guaranteed to use the fewest coins possible?
3. Using a generator of random orderings of the integers $1, 2, \ldots, n$, find the number of comparisons used by the bubble sort, insertion sort, binary insertion sort, and selection sort to sort these integers.

Writing Projects

Respond to these with essays using outside sources.

1. Examine the history of the word *algorithm* and describe the use of this word in early writings.
2. Look up Bachmann's original introduction of big-O notation. Explain how he and others have used this notation.
3. Explain how sorting algorithms can be classified into a taxonomy based on the underlying principle on which they are based.
4. Describe the radix sort algorithm.
5. Describe the historic trends in how quickly processors can perform operations and use these trends to estimate how quickly processors will be able to perform operations in the next twenty years.
6. Develop a detailed list of algorithmic paradigms and provide examples using each of these paradigms.
7. Explain what the Turing Award is and describe the criteria used to select winners. List six past winners of the award and why they received the award.
8. Describe what is meant by a parallel algorithm. Explain how the pseudocode used in this book can be extended to handle parallel algorithms.
9. Explain how the complexity of parallel algorithms can be measured. Give some examples to illustrate this concept, showing how a parallel algorithm can work more quickly than one that does not operate in parallel.
10. Describe six different NP-complete problems.
11. Demonstrate how one of the many different NP-complete problems can be reduced to the satisfiability problem.

CHAPTER

4 Number Theory and Cryptography

The part of mathematics devoted to the study of the set of integers and their properties is known as number theory. In this chapter we will develop some of the important concepts of number theory including many of those used in computer science. As we develop number theory, we will use the proof methods developed in Chapter 1 to prove many theorems.

We will first introduce the notion of divisibility of integers, which we use to introduce modular, or clock, arithmetic. Modular arithmetic operates with the remainders of integers when they are divided by a fixed positive integer, called the modulus. We will prove many important results about modular arithmetic which we will use extensively in this chapter.

Integers can be represented with any positive integer b greater than 1 as a base. In this chapter we discuss base b representations of integers and give an algorithm for finding them. In particular, we will discuss binary, octal, and hexadecimal (base 2, 8, and 16) representations. We will describe algorithms for carrying out arithmetic using these representations and study their complexity. These algorithms were the first procedures called algorithms.

We will discuss prime numbers, the positive integers that have only 1 and themselves as positive divisors. We will prove that there are infinitely many primes; the proof we give is considered to be one of the most beautiful proofs in mathematics. We will discuss the distribution of primes and many famous open questions concerning primes. We will introduce the concept of greatest common divisors and study the Euclidean algorithm for computing them. This algorithm was first described thousands of years ago. We will introduce the fundamental theorem of arithmetic, a key result which tells us that every positive integer has a unique factorization into primes.

We will explain how to solve linear congruences, as well as systems of linear congruences, which we solve using the famous Chinese remainder theorem. We will introduce the notion of pseudoprimes, which are composite integers masquerading as primes, and show how this notion can help us rapidly generate prime numbers.

This chapter introduces several important applications of number theory. In particular, we will use number theory to generate pseudorandom numbers, to assign memory locations to computer files, and to find check digits used to detect errors in various kinds of identification numbers. We also introduce the subject of cryptography. Number theory plays an essentially role both in classical cryptography, first used thousands of years ago, and modern cryptography, which plays an essential role in electronic communication. We will show how the ideas we develop can be used in cryptographical protocols, introducing protocols for sharing keys and for sending signed messages. Number theory, once considered the purest of subjects, has become an essential tool in providing computer and Internet security.

4.1 Divisibility and Modular Arithmetic

Introduction

The ideas that we will develop in this section are based on the notion of divisibility. Division of an integer by a positive integer produces a quotient and a remainder. Working with these remainders leads to modular arithmetic, which plays an important role in mathematics and which is used throughout computer science. We will discuss some important applications of modular arithmetic

later in this chapter, including generating pseudorandom numbers, assigning computer memory locations to files, constructing check digits, and encrypting messages.

Division

When one integer is divided by a second nonzero integer, the quotient may or may not be an integer. For example, $12/3 = 4$ is an integer, whereas $11/4 = 2.75$ is not. This leads to Definition 1.

DEFINITION 1 If a and b are integers with $a \neq 0$, we say that a *divides* b if there is an integer c such that $b = ac$, or equivalently, if $\frac{b}{a}$ is an integer. When a divides b we say that a is a *factor* or *divisor* of b, and that b is a *multiple* of a. The notation $a \mid b$ denotes that a divides b. We write $a \nmid b$ when a does not divide b.

Remark: We can express $a \mid b$ using quantifiers as $\exists c(ac = b)$, where the universe of discourse is the set of integers.

In Figure 1 a number line indicates which integers are divisible by the positive integer d.

EXAMPLE 1 Determine whether $3 \mid 7$ and whether $3 \mid 12$.

Solution: We see that $3 \nmid 7$, because $7/3$ is not an integer. On the other hand, $3 \mid 12$ because $12/3 = 4$. ◀

EXAMPLE 2 Let n and d be positive integers. How many positive integers not exceeding n are divisible by d?

Solution: The positive integers divisible by d are all the integers of the form dk, where k is a positive integer. Hence, the number of positive integers divisible by d that do not exceed n equals the number of integers k with $0 < dk \leq n$, or with $0 < k \leq n/d$. Therefore, there are $\lfloor n/d \rfloor$ positive integers not exceeding n that are divisible by d. ◀

Extra Examples

Some of the basic properties of divisibility of integers are given in Theorem 1.

THEOREM 1 Let a, b, and c be integers, where $a \neq 0$. Then

(*i*) if $a \mid b$ and $a \mid c$, then $a \mid (b + c)$;
(*ii*) if $a \mid b$, then $a \mid bc$ for all integers c;
(*iii*) if $a \mid b$ and $b \mid c$, then $a \mid c$.

Proof: We will give a direct proof of (*i*). Suppose that $a \mid b$ and $a \mid c$. Then, from the definition of divisibility, it follows that there are integers s and t with $b = as$ and $c = at$. Hence,

$$b + c = as + at = a(s + t).$$

FIGURE 1 Integers Divisible by the Positive Integer *d*.

Therefore, a divides $b + c$. This establishes part (i) of the theorem. The proofs of parts (ii) and (iii) are left as Exercises 3 and 4. ◁

Theorem 1 has this useful consequence.

COROLLARY 1 If a, b, and c are integers, where $a \neq 0$, such that $a \mid b$ and $a \mid c$, then $a \mid mb + nc$ whenever m and n are integers.

Proof: We will give a direct proof. By part (ii) of Theorem 1 we see that $a \mid mb$ and $a \mid nc$ whenever m and n are integers. By part (i) of Theorem 1 it follows that $a \mid mb + nc$. ◁

The Division Algorithm

When an integer is divided by a positive integer, there is a quotient and a remainder, as the division algorithm shows.

THEOREM 2 **THE DIVISION ALGORITHM** Let a be an integer and d a positive integer. Then there are unique integers q and r, with $0 \leq r < d$, such that $a = dq + r$.

We defer the proof of the division algorithm to Section 5.2. (See Example 5 and Exercise 37.)

Remark: Theorem 2 is not really an algorithm. (Why not?) Nevertheless, we use its traditional name.

DEFINITION 2 In the equality given in the division algorithm, d is called the *divisor*, a is called the *dividend*, q is called the *quotient*, and r is called the *remainder*. This notation is used to express the quotient and remainder:

$$q = a \textbf{ div } d, \quad r = a \textbf{ mod } d.$$

Remark: Note that both a **div** d and a **mod** d for a fixed d are functions on the set of integers. Furthermore, when a is an integer and d is a positive integer, we have a **div** $d = \lfloor a/d \rfloor$ and a **mod** $d = a - d$. (See exercise 18.)

Examples 3 and 4 illustrate the division algorithm.

EXAMPLE 3 What are the quotient and remainder when 101 is divided by 11?

Solution: We have

$$101 = 11 \cdot 9 + 2.$$

Hence, the quotient when 101 is divided by 11 is $9 = 101$ **div** 11, and the remainder is $2 = 101$ **mod** 11. ◀

EXAMPLE 4 What are the quotient and remainder when -11 is divided by 3?

Solution: We have

$$-11 = 3(-4) + 1.$$

Extra Examples

Hence, the quotient when -11 is divided by 3 is $-4 = -11$ **div** 3, and the remainder is $1 = -11$ **mod** 3.

Note that the remainder cannot be negative. Consequently, the remainder is *not* -2, even though

$$-11 = 3(-3) - 2,$$

because $r = -2$ does not satisfy $0 \leq r < 3$. ◀

Note that the integer a is divisible by the integer d if and only if the remainder is zero when a is divided by d.

Remark: A programming language may have one, or possibly two, operators for modular arithmetic, denoted by mod (in BASIC, Maple, Mathematica, EXCEL, and SQL), % (in C, C++, Java, and Python), rem (in Ada and Lisp), or something else. Be careful when using them, because for $a < 0$, some of these operators return $a - m\lceil a/m \rceil$ instead of a **mod** $m = a - m\lfloor a/m \rfloor$ (as shown in Exercise 18). Also, unlike a **mod** m, some of these operators are defined when $m < 0$, and even when $m = 0$.

Modular Arithmetic

In some situations we care only about the remainder of an integer when it is divided by some specified positive integer. For instance, when we ask what time it will be (on a 24-hour clock) 50 hours from now, we care only about the remainder when 50 plus the current hour is divided by 24. Because we are often interested only in remainders, we have special notations for them. We have already introduced the notation a **mod** m to represent the remainder when an integer a is divided by the positive integer m. We now introduce a different, but related, notation that indicates that two integers have the same remainder when they are divided by the positive integer m.

DEFINITION 3 If a and b are integers and m is a positive integer, then a is *congruent to b modulo m* if m divides $a - b$. We use the notation $a \equiv b \pmod{m}$ to indicate that a is congruent to b modulo m. We say that $a \equiv b \pmod{m}$ is a **congruence** and that m is its **modulus** (plural **moduli**). If a and b are not congruent modulo m, we write $a \not\equiv b \pmod{m}$.

Although both notations $a \equiv b \pmod{m}$ and a **mod** $m = b$ include "mod," they represent fundamentally different concepts. The first represents a relation on the set of integers, whereas the second represents a function. However, the relation $a \equiv b \pmod{m}$ and the **mod** m function are closely related, as described in Theorem 3.

THEOREM 3 Let a and b be integers, and let m be a positive integer. Then $a \equiv b \pmod{m}$ if and only if $a \bmod m = b \bmod m$.

The proof of Theorem 3 is left as Exercises 15 and 16. Recall that $a \bmod m$ and $b \bmod m$ are the remainders when a and b are divided by m, respectively. Consequently, Theorem 3 also says that $a \equiv b \pmod{m}$ if and only if a and b have the same remainder when divided by m.

EXAMPLE 5 Determine whether 17 is congruent to 5 modulo 6 and whether 24 and 14 are congruent modulo 6.

Solution: Because 6 divides $17 - 5 = 12$, we see that $17 \equiv 5 \pmod{6}$. However, because $24 - 14 = 10$ is not divisible by 6, we see that $24 \not\equiv 14 \pmod{6}$. ◀

The great German mathematician Karl Friedrich Gauss developed the concept of congruences at the end of the eighteenth century. The notion of congruences has played an important role in the development of number theory.

Theorem 4 provides a useful way to work with congruences.

THEOREM 4 Let m be a positive integer. The integers a and b are congruent modulo m if and only if there is an integer k such that $a = b + km$.

Proof: If $a \equiv b \pmod{m}$, by the definition of congruence (Definition 3), we know that $m \mid (a - b)$. This means that there is an integer k such that $a - b = km$, so that $a = b + km$. Conversely, if there is an integer k such that $a = b + km$, then $km = a - b$. Hence, m divides $a - b$, so that $a \equiv b \pmod{m}$. ◁

The set of all integers congruent to an integer a modulo m is called the **congruence class** of a modulo m. In Chapter 9 we will show that there are m pairwise disjoint equivalence classes modulo m and that the union of these equivalence classes is the set of integers.

Theorem 5 shows that additions and multiplications preserve congruences.

Links

KARL FRIEDRICH GAUSS (1777–1855) Karl Friedrich Gauss, the son of a bricklayer, was a child prodigy. He demonstrated his potential at the age of 10, when he quickly solved a problem assigned by a teacher to keep the class busy. The teacher asked the students to find the sum of the first 100 positive integers. Gauss realized that this sum could be found by forming 50 pairs, each with the sum 101: $1 + 100, 2 + 99, \ldots, 50 + 51$. This brilliance attracted the sponsorship of patrons, including Duke Ferdinand of Brunswick, who made it possible for Gauss to attend Caroline College and the University of Göttingen. While a student, he invented the method of least squares, which is used to estimate the most likely value of a variable from experimental results. In 1796 Gauss made a fundamental discovery in geometry, advancing a subject that had not advanced since ancient times. He showed that a 17-sided regular polygon could be drawn using just a ruler and compass.

In 1799 Gauss presented the first rigorous proof of the fundamental theorem of algebra, which states that a polynomial of degree n has exactly n roots (counting multiplicities). Gauss achieved worldwide fame when he successfully calculated the orbit of the first asteroid discovered, Ceres, using scanty data.

Gauss was called the Prince of Mathematics by his contemporary mathematicians. Although Gauss is noted for his many discoveries in geometry, algebra, analysis, astronomy, and physics, he had a special interest in number theory, which can be seen from his statement "Mathematics is the queen of the sciences, and the theory of numbers is the queen of mathematics." Gauss laid the foundations for modern number theory with the publication of his book *Disquisitiones Arithmeticae* in 1801.

THEOREM 5

Let m be a positive integer. If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, then

$$a + c \equiv b + d \pmod{m} \qquad \text{and} \qquad ac \equiv bd \pmod{m}.$$

Proof: We use a direct proof. Because $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, by Theorem 4 there are integers s and t with $b = a + sm$ and $d = c + tm$. Hence,

$$b + d = (a + sm) + (c + tm) = (a + c) + m(s + t)$$

and

$$bd = (a + sm)(c + tm) = ac + m(at + cs + stm).$$

Hence,

$$a + c \equiv b + d \pmod{m} \qquad \text{and} \qquad ac \equiv bd \pmod{m}.$$ ◁

EXAMPLE 6 Because $7 \equiv 2 \pmod 5$ and $11 \equiv 1 \pmod 5$, it follows from Theorem 5 that

$$18 = 7 + 11 \equiv 2 + 1 = 3 \pmod 5$$

and that

$$77 = 7 \cdot 11 \equiv 2 \cdot 1 = 2 \pmod 5.$$ ◀

You cannot always divide both sides of a congruence by the same number!

We must be careful working with congruences. Some properties we may expect to be true are not valid. For example, if $ac \equiv bc \pmod m$, the congruence $a \equiv b \pmod m$ may be false. Similarly, if $a \equiv b \pmod m$ and $c \equiv d \pmod m$, the congruence $a^c \equiv b^d \pmod m$ may be false. (See Exercise 37.)

Corollary 2 shows how to find the values of the **mod** m function at the sum and product of two integers using the values of this function at each of these integers. We will use this result in Section 5.4.

COROLLARY 2

Let m be a positive integer and let a and b be integers. Then

$$(a + b) \bmod m = ((a \bmod m) + (b \bmod m)) \bmod m$$

and

$$ab \bmod m = ((a \bmod m)(b \bmod m)) \bmod m.$$

Proof: By the definitions of **mod** m and of congruence modulo m, we know that $a \equiv (a \bmod m) \pmod m$ and $b \equiv (b \bmod m) \pmod m$. Hence, Theorem 5 tells us that

$$a + b \equiv (a \bmod m) + (b \bmod m) \pmod m$$

and

$$ab \equiv (a \bmod m)(b \bmod m) \pmod m.$$

The equalities in this corollary follow from these last two congruences by Theorem 3. ◁

Arithmetic Modulo m

We can define arithmetic operations on $\mathbf{Z}_m$, the set of nonnegative integers less than m, that is, the set $\{0, 1, \ldots, m-1\}$. In particular, we define addition of these integers, denoted by $+_m$ by

$$a +_m b = (a + b) \bmod m,$$

where the addition on the right-hand side of this equation is the ordinary addition of integers, and we define multiplication of these integers, denoted by $\cdot_m$ by

$$a \cdot_m b = (a \cdot b) \bmod m,$$

where the multiplication on the right-hand side of this equation is the ordinary multiplication of integers. The operations $+_m$ and $\cdot_m$ are called addition and multiplication modulo m and when we use these operations, we are said to be doing **arithmetic modulo** m.

EXAMPLE 7 Use the definition of addition and multiplication in $\mathbf{Z}_m$ to find $7 +_{11} 9$ and $7 \cdot_{11} 9$.

Solution: Using the definition of addition modulo 11, we find that

$$7 +_{11} 9 = (7 + 9) \bmod 11 = 16 \bmod 11 = 5,$$

and

$$7 \cdot_{11} 9 = (7 \cdot 9) \bmod 11 = 63 \bmod 11 = 8.$$

Hence $7 +_{11} 9 = 5$ and $7 \cdot_{11} 9 = 8$. ◀

The operations $+_m$ and $\cdot_m$ satisfy many of the same properties of ordinary addition and multiplication of integers. In particular, they satisfy these properties:

Closure If a and b belong to $\mathbf{Z}_m$, then $a +_m b$ and $a \cdot_m b$ belong to $\mathbf{Z}_m$.

Associativity If a, b, and c belong to $\mathbf{Z}_m$, then $(a +_m b) +_m c = a +_m (b +_m c)$ and $(a \cdot_m b) \cdot_m c = a \cdot_m (b \cdot_m c)$.

Commutativity If a and b belong to $\mathbf{Z}_m$, then $a +_m b = b +_m a$ and $a \cdot_m b = b \cdot_m a$.

Identity elements The elements 0 and 1 are identity elements for addition and multiplication modulo m, respectively. That is, if a belongs to $\mathbf{Z}_m$, then $a +_m 0 = 0 +_m a = a$ and $a \cdot_m 1 = 1 \cdot_m a = a$.

Additive inverses If $a \neq 0$ belongs to $\mathbf{Z}_m$, then $m - a$ is an additive inverse of a modulo m and 0 is its own additive inverse. That is $a +_m (m - a) = 0$ and $0 +_m 0 = 0$.

Distributivity If a, b, and c belong to $\mathbf{Z}_m$, then $a \cdot_m (b +_m c) = (a \cdot_m b) +_m (a \cdot_m c)$ and $(a +_m b) \cdot_m c = (a \cdot_m c) +_m (b \cdot_m c)$.

These properties follow from the properties we have developed for congruences and remainders modulo m, together with the properties of integers; we leave their proofs as Exercises 42–44. Note that we have listed the property that every element of $\mathbf{Z}_m$ has an additive inverse, but no analogous property for multiplicative inverses has been included. This is because multiplicative inverses do not always exists modulo m. For instance, there is no multiplicative inverse of 2 modulo 6, as the reader can verify. We will return to the question of when an integer has a multiplicative inverse modulo m later in this chapter.

Remark: Because $\mathbf{Z}_m$ with the operations of addition and multiplication modulo m satisfies the properties listed, $\mathbf{Z}_m$ with modular addition is said to be a **commutative group** and $\mathbf{Z}_m$ with both of these operations is said to be a **commutative ring**. Note that the set of integers with ordinary addition and multiplication also forms a commutative ring. Groups and rings are studied in courses that cover abstract algebra.

Remark: In Exercise 30, and in later sections, we will use the notations $+$ and $\cdot$ for $+_m$ and $\cdot_m$ without the subscript m on the symbol for the operator whenever we work with $\mathbf{Z}_m$.

Exercises

1. Does 17 divide each of these numbers?
a) 68 **b)** 84 **c)** 357 **d)** 1001

2. Prove that if a is an integer other than 0, then
a) 1 divides a. **b)** a divides 0.

3. Prove that part (*ii*) of Theorem 1 is true.

4. Prove that part (*iii*) of Theorem 1 is true.

5. Show that if $a \mid b$ and $b \mid a$, where a and b are integers, then $a = b$ or $a = -b$.

6. Show that if a, b, c, and d are integers, where $a \neq 0$, such that $a \mid c$ and $b \mid d$, then $ab \mid cd$.

7. Show that if a, b, and c are integers, where $a \neq 0$ and $c \neq 0$, such that $ac \mid bc$, then $a \mid b$.

8. Prove or disprove that if $a \mid bc$, where a, b, and c are positive integers and $a \neq 0$, then $a \mid b$ or $a \mid c$.

9. What are the quotient and remainder when
a) 19 is divided by 7?
b) -111 is divided by 11?
c) 789 is divided by 23?
d) 1001 is divided by 13?
e) 0 is divided by 19?
f) 3 is divided by 5?
g) -1 is divided by 3?
h) 4 is divided by 1?

10. What are the quotient and remainder when
a) 44 is divided by 8?
b) 777 is divided by 21?
c) -123 is divided by 19?
d) -1 is divided by 23?
e) -2002 is divided by 87?
f) 0 is divided by 17?
g) 1,234,567 is divided by 1001?
h) -100 is divided by 101?

11. What time does a 12-hour clock read
a) 80 hours after it reads 11:00?
b) 40 hours before it reads 12:00?
c) 100 hours after it reads 6:00?

12. What time does a 24-hour clock read
a) 100 hours after it reads 2:00?
b) 45 hours before it reads 12:00?
c) 168 hours after it reads 19:00?

13. Suppose that a and b are integers, $a \equiv 4 \pmod{13}$, and $b \equiv 9 \pmod{13}$. Find the integer c with $0 \leq c \leq 12$ such that
a) $c \equiv 9a \pmod{13}$.
b) $c \equiv 11b \pmod{13}$.
c) $c \equiv a + b \pmod{13}$.
d) $c \equiv 2a + 3b \pmod{13}$.
e) $c \equiv a^2 + b^2 \pmod{13}$.
f) $c \equiv a^3 - b^3 \pmod{13}$.

14. Suppose that a and b are integers, $a \equiv 11 \pmod{19}$, and $b \equiv 3 \pmod{19}$. Find the integer c with $0 \leq c \leq 18$ such that
a) $c \equiv 13a \pmod{19}$.
b) $c \equiv 8b \pmod{19}$.
c) $c \equiv a - b \pmod{19}$.
d) $c \equiv 7a + 3b \pmod{19}$.
e) $c \equiv 2a^2 + 3b^2 \pmod{19}$.
f) $c \equiv a^3 + 4b^3 \pmod{19}$.

15. Let m be a positive integer. Show that $a \equiv b \pmod{m}$ if $a \bmod m = b \bmod m$.

16. Let m be a positive integer. Show that $a \bmod m = b \bmod m$ if $a \equiv b \pmod{m}$.

17. Show that if n and k are positive integers, then $\lceil n/k \rceil = \lfloor (n-1)/k \rfloor + 1$.

18. Show that if a is an integer and d is an integer greater than 1, then the quotient and remainder obtained when a is divided by d are $\lfloor a/d \rfloor$ and $a - d\lfloor a/d \rfloor$, respectively.

19. Find a formula for the integer with smallest absolute value that is congruent to an integer a modulo m, where m is a positive integer.

20. Evaluate these quantities.
a) $-17 \bmod 2$ **b)** $144 \bmod 7$
c) $-101 \bmod 13$ **d)** $199 \bmod 19$

21. Evaluate these quantities.
a) $13 \bmod 3$ **b)** $-97 \bmod 11$
c) $155 \bmod 19$ **d)** $-221 \bmod 23$

22. Find $a \operatorname{div} m$ and $a \bmod m$ when
a) $a = -111, m = 99$.
b) $a = -9999, m = 101$.
c) $a = 10299, m = 999$.
d) $a = 123456, m = 1001$.

23. Find a **div** m and a **mod** m when
  a) $a = 228, m = 119$.
  b) $a = 9009, m = 223$.
  c) $a = -10101, m = 333$.
  d) $a = -765432, m = 38271$.

24. Find the integer a such that
  a) $a \equiv 43 \pmod{23}$ and $-22 \leq a \leq 0$.
  b) $a \equiv 17 \pmod{29}$ and $-14 \leq a \leq 14$.
  c) $a \equiv -11 \pmod{21}$ and $90 \leq a \leq 110$.

25. Find the integer a such that
  a) $a \equiv -15 \pmod{27}$ and $-26 \leq a \leq 0$.
  b) $a \equiv 24 \pmod{31}$ and $-15 \leq a \leq 15$.
  c) $a \equiv 99 \pmod{41}$ and $100 \leq a \leq 140$.

26. List five integers that are congruent to 4 modulo 12.

27. List all integers between -100 and 100 that are congruent to -1 modulo 25.

28. Decide whether each of these integers is congruent to 3 modulo 7.
  a) 37 **b)** 66
  c) -17 **d)** -67

29. Decide whether each of these integers is congruent to 5 modulo 17.
  a) 80 **b)** 103
  c) -29 **d)** -122

30. Find each of these values.
  a) $(177 \textbf{ mod } 31 + 270 \textbf{ mod } 31) \textbf{ mod } 31$
  b) $(177 \textbf{ mod } 31 \cdot 270 \textbf{ mod } 31) \textbf{ mod } 31$

31. Find each of these values.
  a) $(-133 \textbf{ mod } 23 + 261 \textbf{ mod } 23) \textbf{ mod } 23$
  b) $(457 \textbf{ mod } 23 \cdot 182 \textbf{ mod } 23) \textbf{ mod } 23$

32. Find each of these values.
  a) $(19^2 \textbf{ mod } 41) \textbf{ mod } 9$
  b) $(32^3 \textbf{ mod } 13)^2 \textbf{ mod } 11$
  c) $(7^3 \textbf{ mod } 23)^2 \textbf{ mod } 31$
  d) $(21^2 \textbf{ mod } 15)^3 \textbf{ mod } 22$

33. Find each of these values.
  a) $(99^2 \textbf{ mod } 32)^3 \textbf{ mod } 15$
  b) $(3^4 \textbf{ mod } 17)^2 \textbf{ mod } 11$
  c) $(19^3 \textbf{ mod } 23)^2 \textbf{ mod } 31$
  d) $(89^3 \textbf{ mod } 79)^4 \textbf{ mod } 26$

34. Show that if $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, where a, b, c, d, and m are integers with $m \geq 2$, then $a - c \equiv b - d \pmod{m}$.

35. Show that if $n \mid m$, where n and m are integers greater than 1, and if $a \equiv b \pmod{m}$, where a and b are integers, then $a \equiv b \pmod{n}$.

☞ **36.** Show that if a, b, c, and m are integers such that $m \geq 2$, $c > 0$, and $a \equiv b \pmod{m}$, then $ac \equiv bc \pmod{mc}$.

37. Find counterexamples to each of these statements about congruences.
  a) If $ac \equiv bc \pmod{m}$, where a, b, c, and m are integers with $m \geq 2$, then $a \equiv b \pmod{m}$.
  b) If $a \equiv b \pmod{m}$ and $c \equiv d \pmod{m}$, where a, b, c, d, and m are integers with c and d positive and $m \geq 2$, then $a^c \equiv b^d \pmod{m}$.

38. Show that if n is an integer then $n^2 \equiv 0$ or $1 \pmod{4}$.

39. Use Exercise 38 to show that if m is a positive integer of the form $4k + 3$ for some nonnegative integer k, then m is not the sum of the squares of two integers.

40. Prove that if n is an odd positive integer, then $n^2 \equiv 1 \pmod{8}$.

41. Show that if a, b, k, and m are integers such that $k \geq 1$, $m \geq 2$, and $a \equiv b \pmod{m}$, then $a^k \equiv b^k \pmod{m}$.

42. Show that $\mathbf{Z}_m$ with addition modulo m, where $m \geq 2$ is an integer, satisfies the closure, associative, and commutative properties, 0 is an additive identity, and for every nonzero $a \in \mathbf{Z}_m$, $m - a$ is an inverse of a modulo m.

43. Show that $\mathbf{Z}_m$ with multiplication modulo m, where $m \geq 2$ is an integer, satisfies the closure, associative, and commutativity properties, and 1 is a multiplicative identity.

44. Show that the distributive property of multiplication over addition holds for $\mathbf{Z}_m$, where $m \geq 2$ is an integer.

45. Write out the addition and multiplication tables for $\mathbf{Z}_5$ (where by addition and multiplication we mean $+_5$ and $\cdot_5$).

46. Write out the addition and multiplication tables for $\mathbf{Z}_6$ (where by addition and multiplication we mean $+_6$ and $\cdot_6$).

47. Determine whether each of the functions $f(a) = a$ **div** d and $g(a) = a$ **mod** d, where d is a fixed positive integer, from the set of integers to the set of integers, is one-to-one, and determine whether each of these functions is onto.

4.2 Integer Representations and Algorithms

Introduction

Integers can be expressed using any integer greater than one as a base, as we will show in this section. Although we commonly use decimal (base 10), representations, binary (base 2), octal (base 8), and hexadecimal (base 16) representations are often used, especially in computer science. Given a base b and an integer n, we will show how to construct the base b representation of this integer. We will also explain how to quickly covert between binary and octal and between binary and hexadecimal notations.

As mentioned in Section 3.1, the term *algorithm* originally referred to procedures for performing arithmetic operations using the decimal representations of integers. These algorithms, adapted for use with binary representations, are the basis for computer arithmetic. They provide good illustrations of the concept of an algorithm and the complexity of algorithms. For these reasons, they will be discussed in this section.

We will also introduce an algorithm for finding a **div** d and a **mod** d where a and d are integers with $d > 1$. Finally, we will describe an efficient algorithm for modular exponentiation, which is a particularly important algorithm for cryptography, as we will see in Section 4.6.

Representations of Integers

In everyday life we use decimal notation to express integers. For example, 965 is used to denote $9 \cdot 10^2 + 6 \cdot 10 + 5$. However, it is often convenient to use bases other than 10. In particular, computers usually use binary notation (with 2 as the base) when carrying out arithmetic, and octal (base 8) or hexadecimal (base 16) notation when expressing characters, such as letters or digits. In fact, we can use any integer greater than 1 as the base when expressing integers. This is stated in Theorem 1.

THEOREM 1

Let b be an integer greater than 1. Then if n is a positive integer, it can be expressed uniquely in the form

$$n = a_k b^k + a_{k-1} b^{k-1} + \cdots + a_1 b + a_0,$$

where k is a nonnegative integer, $a_0, a_1, \ldots, a_k$ are nonnegative integers less than b, and $a_k \neq 0$.

A proof of this theorem can be constructed using mathematical induction, a proof method that is discussed in Section 5.1. It can also be found in [Ro10]. The representation of n given in Theorem 1 is called the **base b expansion of n**. The base b expansion of n is denoted by $(a_k a_{k-1} \ldots a_1 a_0)_b$. For instance, $(245)_8$ represents $2 \cdot 8^2 + 4 \cdot 8 + 5 = 165$. Typically, the subscript 10 is omitted for base 10 expansions of integers because base 10, or **decimal expansions**, are commonly used to represent integers.

BINARY EXPANSIONS Choosing 2 as the base gives **binary expansions** of integers. In binary notation each digit is either a 0 or a 1. In other words, the binary expansion of an integer is just a bit string. Binary expansions (and related expansions that are variants of binary expansions) are used by computers to represent and do arithmetic with integers.

EXAMPLE 1 What is the decimal expansion of the integer that has $(1\ 0101\ 1111)_2$ as its binary expansion?

Solution: We have

$$\begin{aligned}(1\ 0101\ 1111)_2 = {} & 1 \cdot 2^8 + 0 \cdot 2^7 + 1 \cdot 2^6 + 0 \cdot 2^5 + 1 \cdot 2^4 \\ & + 1 \cdot 2^3 + 1 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 351.\end{aligned}$$

◀

OCTAL AND HEXADECIMAL EXPANSIONS Among the most important bases in computer science are base 2, base 8, and base 16. Base 8 expansions are called **octal** expansions and base 16 expansions are **hexadecimal** expansions.

EXAMPLE 2 What is the decimal expansion of the number with octal expansion $(7016)_8$?

Solution: Using the definition of a base b expansion with $b = 8$ tells us that

$$(7016)_8 = 7 \cdot 8^3 + 0 \cdot 8^2 + 1 \cdot 8 + 6 = 3598.$$ ◀

Sixteen different digits are required for hexadecimal expansions. Usually, the hexadecimal digits used are 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F, where the letters A through F represent the digits corresponding to the numbers 10 through 15 (in decimal notation).

EXAMPLE 3 What is the decimal expansion of the number with hexadecimal expansion $(2AE0B)_{16}$?

Solution: Using the definition of a base b expansion with $b = 16$ tells us that

$$(2AE0B)_{16} = 2 \cdot 16^4 + 10 \cdot 16^3 + 14 \cdot 16^2 + 0 \cdot 16 + 11 = 175627.$$ ◀

Each hexadecimal digit can be represented using four bits. For instance, we see that $(1110\ \ 0101)_2 = (E5)_{16}$ because $(1110)_2 = (E)_{16}$ and $(0101)_2 = (5)_{16}$. **Bytes**, which are bit strings of length eight, can be represented by two hexadecimal digits.

BASE CONVERSION We will now describe an algorithm for constructing the base b expansion of an integer n. First, divide n by b to obtain a quotient and remainder, that is,

$$n = bq_0 + a_0, \qquad 0 \leq a_0 < b.$$

The remainder, a_0, is the rightmost digit in the base b expansion of n. Next, divide q_0 by b to obtain

$$q_0 = bq_1 + a_1, \qquad 0 \leq a_1 < b.$$

We see that a_1 is the second digit from the right in the base b expansion of n. Continue this process, successively dividing the quotients by b, obtaining additional base b digits as the remainders. This process terminates when we obtain a quotient equal to zero. It produces the base b digits of n from the right to the left.

EXAMPLE 4 Find the octal expansion of $(12345)_{10}$.

Extra Examples

Solution: First, divide 12345 by 8 to obtain

$$12345 = 8 \cdot 1543 + 1.$$

Successively dividing quotients by 8 gives

$$\begin{aligned} 1543 &= 8 \cdot 192 + 7, \\ 192 &= 8 \cdot 24 + 0, \\ 24 &= 8 \cdot 3 + 0, \\ 3 &= 8 \cdot 0 + 3. \end{aligned}$$

The successive remainders that we have found, 1, 7, 0, 0, and 3, are the digits from the right to the left of 12345 in base 8. Hence,

$$(12345)_{10} = (30071)_8.$$ ◀

EXAMPLE 5 Find the hexadecimal expansion of $(177130)_{10}$.

Solution: First divide 177130 by 16 to obtain

$$177130 = 16 \cdot 11070 + 10.$$

Successively dividing quotients by 16 gives

$$\begin{aligned} 11070 &= 16 \cdot 691 + 14, \\ 691 &= 16 \cdot 43 + 3, \\ 43 &= 16 \cdot 2 + 11, \\ 2 &= 16 \cdot 0 + 2. \end{aligned}$$

The successive remainders that we have found, 10, 14, 3, 11, 2, give us the digits from the right to the left in the hexadecimal (base 16) expansion of $(177130)_{10}$. It follows that

$$(177130)_{10} = (2B3EA)_{16}.$$

(Recall that the integers 10, 11, and 14 correspond to the hexadecimal digits A, B, and E, respectively.) ◀

EXAMPLE 6 Find the binary expansion of $(241)_{10}$.

Solution: First divide 241 by 2 to obtain

$$241 = 2 \cdot 120 + 1.$$

Successively dividing quotients by 2 gives

$$\begin{aligned} 120 &= 2 \cdot 60 + 0, \\ 60 &= 2 \cdot 30 + 0, \\ 30 &= 2 \cdot 15 + 0, \\ 15 &= 2 \cdot 7 + 1, \\ 7 &= 2 \cdot 3 + 1, \\ 3 &= 2 \cdot 1 + 1, \\ 1 &= 2 \cdot 0 + 1. \end{aligned}$$

The successive remainders that we have found, 1, 0, 0, 0, 1, 1, 1, 1, are the digits from the right to the left in the binary (base 2) expansion of $(241)_{10}$. Hence,

$$(241)_{10} = (1111\ 0001)_2.$$ ◀

The pseudocode given in Algorithm 1 finds the base b expansion $(a_{k-1} \ldots a_1 a_0)_b$ of the integer n.

TABLE 1 Hexadecimal, Octal, and Binary Representation of the Integers 0 through 15.

Decimal	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hexadecimal	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Octal	0	1	2	3	4	5	6	7	10	11	12	13	14	15	16	17
Binary	0	1	10	11	100	101	110	111	1000	1001	1010	1011	1100	1101	1110	1111

ALGORITHM 1 **Constructing Base b Expansions.**

procedure *base b expansion*(n, b: positive integers with $b > 1$)
$q := n$
$k := 0$
while $q \neq 0$
    $a_k := q$ **mod** b
    $q := q$ **div** b
    $k := k + 1$
return $(a_{k-1}, \ldots, a_1, a_0)$ {$(a_{k-1} \ldots a_1 a_0)_b$ is the base b expansion of n}

In Algorithm 1, q represents the quotient obtained by successive divisions by b, starting with $q = n$. The digits in the base b expansion are the remainders of these divisions and are given by q **mod** b. The algorithm terminates when a quotient $q = 0$ is reached.

Remark: Note that Algorithm 1 can be thought of as a greedy algorithm, as the base b digits are taken as large as possible in each step.

CONVERSION BETWEEN BINARY, OCTAL, AND HEXADECIMAL EXPANSIONS Conversion between binary and octal and between binary and hexadecimal expansions is extremely easy because each octal digit corresponds to a block of three binary digits and each hexadecimal digit corresponds to a block of four binary digits, with these correspondences shown in Table 1 without initial 0s shown. (We leave it as Exercises 13–16 to show that this is the case.) This conversion is illustrated in Example 7.

EXAMPLE 7 Find the octal and hexadecimal expansions of $(11\ 1110\ 1011\ 1100)_2$ and the binary expansions of $(765)_8$ and $(A8D)_{16}$.

Solution: To convert $(11\ 1110\ 1011\ 1100)_2$ into octal notation we group the binary digits into blocks of three, adding initial zeros at the start of the leftmost block if necessary. These blocks, from left to right, are 011, 111, 010, 111, and 100, corresponding to 3, 7, 2, 7, and 4, respectively. Consequently, $(11\ 1110\ 1011\ 1100)_2 = (37274)_8$. To convert $(11\ 1110\ 1011\ 1100)_2$ into hexadecimal notation we group the binary digits into blocks of four, adding initial zeros at the start of the leftmost block if necessary. These blocks, from left to right, are 0011, 1110, 1011, and 1100, corresponding to the hexadecimal digits 3, E, B, and C, respectively. Consequently, $(11\ 1110\ 1011\ 1100)_2 = (3EBC)_{16}$.

To convert $(765)_8$ into binary notation, we replace each octal digit by a block of three binary digits. These blocks are 111, 110, and 101. Hence, $(765)_8 = (1\ 1111\ 0101)_2$. To convert $(A8D)_{16}$ into binary notation, we replace each hexadecimal digit by a block of four binary digits. These blocks are 1010, 1000, and 1101. Hence, $(A8D)_{16} = (1010\ 1000\ 1101)_2$. ◀

Algorithms for Integer Operations

The algorithms for performing operations with integers using their binary expansions are extremely important in computer arithmetic. We will describe algorithms for the addition and the multiplication of two integers expressed in binary notation. We will also analyze the computational complexity of these algorithms, in terms of the actual number of bit operations used. Throughout this discussion, suppose that the binary expansions of a and b are

$$a = (a_{n-1}a_{n-2}\ldots a_1a_0)_2,\ b = (b_{n-1}b_{n-2}\ldots b_1b_0)_2,$$

so that a and b each have n bits (putting bits equal to 0 at the beginning of one of these expansions if necessary).

We will measure the complexity of algorithms for integer arithmetic in terms of the number of bits in these numbers.

ADDITION ALGORITHM Consider the problem of adding two integers in binary notation. A procedure to perform addition can be based on the usual method for adding numbers with pencil and paper. This method proceeds by adding pairs of binary digits together with carries, when they occur, to compute the sum of two integers. This procedure will now be specified in detail.

To add a and b, first add their rightmost bits. This gives

$$a_0 + b_0 = c_0 \cdot 2 + s_0,$$

where s_0 is the rightmost bit in the binary expansion of $a + b$ and c_0 is the **carry**, which is either 0 or 1. Then add the next pair of bits and the carry,

$$a_1 + b_1 + c_0 = c_1 \cdot 2 + s_1,$$

where s_1 is the next bit (from the right) in the binary expansion of $a + b$, and c_1 is the carry. Continue this process, adding the corresponding bits in the two binary expansions and the carry, to determine the next bit from the right in the binary expansion of $a + b$. At the last stage, add a_{n-1}, b_{n-1}, and c_{n-2} to obtain $c_{n-1} \cdot 2 + s_{n-1}$. The leading bit of the sum is $s_n = c_{n-1}$. This procedure produces the binary expansion of the sum, namely, $a + b = (s_n s_{n-1} s_{n-2} \ldots s_1 s_0)_2$.

EXAMPLE 8 Add $a = (1110)_2$ and $b = (1011)_2$.

Solution: Following the procedure specified in the algorithm, first note that

$$a_0 + b_0 = 0 + 1 = 0 \cdot 2 + 1,$$

so that $c_0 = 0$ and $s_0 = 1$. Then, because

$$a_1 + b_1 + c_0 = 1 + 1 + 0 = 1 \cdot 2 + 0,$$

it follows that $c_1 = 1$ and $s_1 = 0$. Continuing,

$$a_2 + b_2 + c_1 = 1 + 0 + 1 = 1 \cdot 2 + 0,$$

so that $c_2 = 1$ and $s_2 = 0$. Finally, because

$$a_3 + b_3 + c_2 = 1 + 1 + 1 = 1 \cdot 2 + 1,$$

follows that $c_3 = 1$ and $s_3 = 1$. This means that $s_4 = c_3 = 1$. Therefore, $s = a + b = (1\,1001)_2$. This addition is displayed in Figure 1, where carries are shown in blue. ◀

```
 1 1 1
   1 1 1 0
 + 1 0 1 1
 ---------
 1 1 0 0 1
```

FIGURE 1 **Adding $(1110)_2$ and $(1011)_2$.**

The algorithm for addition can be described using pseudocode as follows.

ALGORITHM 2 Addition of Integers.

procedure *add*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1}a_{n-2}\ldots a_1a_0)_2$ and $(b_{n-1}b_{n-2}\ldots b_1b_0)_2$, respectively}
$c := 0$
for $j := 0$ **to** $n - 1$
  $d := \lfloor (a_j + b_j + c)/2 \rfloor$
  $s_j := a_j + b_j + c - 2d$
  $c := d$
$s_n := c$
return $(s_0, s_1, \ldots, s_n)$ {the binary expansion of the sum is $(s_n s_{n-1}\ldots s_0)_2$}

Next, the number of additions of bits used by Algorithm 2 will be analyzed.

EXAMPLE 9 How many additions of bits are required to use Algorithm 2 to add two integers with n bits (or less) in their binary representations?

Solution: Two integers are added by successively adding pairs of bits and, when it occurs, a carry. Adding each pair of bits and the carry requires two additions of bits. Thus, the total number of additions of bits used is less than twice the number of bits in the expansion. Hence, the number of additions of bits used by Algorithm 2 to add two n-bit integers is $O(n)$. ◀

MULTIPLICATION ALGORITHM Next, consider the multiplication of two n-bit integers a and b. The conventional algorithm (used when multiplying with pencil and paper) works as follows. Using the distributive law, we see that

$$\begin{aligned} ab &= a(b_0 2^0 + b_1 2^1 + \cdots + b_{n-1} 2^{n-1}) \\ &= a(b_0 2^0) + a(b_1 2^1) + \cdots + a(b_{n-1} 2^{n-1}). \end{aligned}$$

We can compute ab using this equation. We first note that $ab_j = a$ if $b_j = 1$ and $ab_j = 0$ if $b_j = 0$. Each time we multiply a term by 2, we shift its binary expansion one place to the left and add a zero at the tail end of the expansion. Consequently, we can obtain $(ab_j)2^j$ by **shifting** the binary expansion of ab_j j places to the left, adding j zero bits at the tail end of this binary expansion. Finally, we obtain ab by adding the n integers $ab_j 2^j$, $j = 0, 1, 2, \ldots, n - 1$.

Algorithm 3 displays this procedure for multiplication.

ALGORITHM 3 Multiplication of Integers.

procedure *multiply*(a, b: positive integers)
{the binary expansions of a and b are $(a_{n-1}a_{n-2}\ldots a_1a_0)_2$
and $(b_{n-1}b_{n-2}\ldots b_1b_0)_2$, respectively}
for $j := 0$ **to** $n-1$
    if $b_j = 1$ **then** $c_j := a$ shifted j places
    else $c_j := 0$
{$c_0, c_1, \ldots, c_{n-1}$ are the partial products}
$p := 0$
for $j := 0$ **to** $n-1$
    $p := p + c_j$
return p {p is the value of ab}

Example 10 illustrates the use of this algorithm.

EXAMPLE 10 Find the product of $a = (110)_2$ and $b = (101)_2$.

Solution: First note that

$$ab_0 \cdot 2^0 = (110)_2 \cdot 1 \cdot 2^0 = (110)_2,$$
$$ab_1 \cdot 2^1 = (110)_2 \cdot 0 \cdot 2^1 = (0000)_2,$$

and

$$ab_2 \cdot 2^2 = (110)_2 \cdot 1 \cdot 2^2 = (11000)_2.$$

To find the product, add $(110)_2$, $(0000)_2$, and $(11000)_2$. Carrying out these additions (using Algorithm 2, including initial zero bits when necessary) shows that $ab = (1\ 1110)_2$. This multiplication is displayed in Figure 2. ◀

```
  1 1 0
× 1 0 1
-------
  1 1 0
0 0 0
1 1 0
-------
1 1 1 1 0
```

FIGURE 2 Multiplying $(110)_2$ and $(101)_2$.

Next, we determine the number of additions of bits and shifts of bits used by Algorithm 3 to multiply two integers.

EXAMPLE 11 How many additions of bits and shifts of bits are used to multiply a and b using Algorithm 3?

Solution: Algorithm 3 computes the products of a and b by adding the partial products $c_0, c_1, c_2, \ldots$, and c_{n-1}. When $b_j = 1$, we compute the partial product c_j by shifting the binary expansion of a by j bits. When $b_j = 0$, no shifts are required because $c_j = 0$. Hence, to find all n of the integers ab_j2^j, $j = 0, 1, \ldots, n-1$, requires at most

$$0 + 1 + 2 + \cdots + n - 1$$

shifts. Hence, by Example 5 in Section 3.2 the number of shifts required is $O(n^2)$.

To add the integers ab_j from $j = 0$ to $j = n-1$ requires the addition of an n-bit integer, an $(n+1)$-bit integer, ..., and a $(2n)$-bit integer. We know from Example 9 that each of these additions requires $O(n)$ additions of bits. Consequently, a total of $O(n^2)$ additions of bits are required for all n additions. ◀

Surprisingly, there are more efficient algorithms than the conventional algorithm for multiplying integers. One such algorithm, which uses $O(n^{1.585})$ bit operations to multiply n-bit numbers, will be described in Section 8.3.

ALGORITHM FOR div AND mod Given integers a and d, $d > 0$, we can find $q = a$ **div** d and $r = a$ **mod** d using Algorithm 4. In this brute-force algorithm, when a is positive we subtract d from a as many times as necessary until what is left is less than d. The number of times we perform this subtraction is the quotient and what is left over after all these subtractions is the remainder. Algorithm 4 also covers the case where a is negative. This algorithm finds the quotient q and remainder r when $|a|$ is divided by d. Then, when $a < 0$ and $r > 0$, it uses these to find the quotient $-(q + 1)$ and remainder $d - r$ when a is divided by d. We leave it to the reader (Exercise 59) to show that, assuming that $a > d$, this algorithm uses $O(q \log a)$ bit operations.

ALGORITHM 4 Computing div and mod.

procedure *division algorithm*(a: integer, d: positive integer)
$q := 0$
$r := |a|$
while $r \geq d$
    $r := r - d$
    $q := q + 1$
if $a < 0$ and $r > 0$ **then**
    $r := d - r$
    $q := -(q + 1)$
return (q, r) {$q = a$ **div** d is the quotient, $r = a$ **mod** d is the remainder}

There are more efficient algorithms than Algorithm 4 for determining the quotient $q = a$ **div** d and the remainder $r = a$ **mod** d when a positive integer a is divided by a positive integer d (see [Kn98] for details). These algorithms require $O(\log a \cdot \log d)$ bit operations. If both of the binary expansions of a and d contain n or fewer bits, then we can replace $\log a \cdot \log d$ by n^2. This means that we need $O(n^2)$ bit operations to find the quotient and remainder when a is divided by d.

Modular Exponentiation

In cryptography it is important to be able to find b^n **mod** m efficiently, where b, n, and m are large integers. It is impractical to first compute b^n and then find its remainder when divided by m because b^n will be a huge number. Instead, we can use an algorithm that employs the binary expansion of the exponent n.

Before we present this algorithm, we illustrate its basic idea. We will explain how to use the binary expansion of n, say $n = (a_{k-1} \ldots a_1 a_0)_2$, to compute b^n. First, note that

$$b^n = b^{a_{k-1} \cdot 2^{k-1} + \cdots + a_1 \cdot 2 + a_0} = b^{a_{k-1} \cdot 2^{k-1}} \cdots b^{a_1 \cdot 2} \cdot b^{a_0}.$$

This shows that to compute b^n, we need only compute the values of b, b^2, $(b^2)^2 = b^4$, $(b^4)^2 = b^8$, $\ldots$, b^{2^k}. Once we have these values, we multiply the terms b^{2^j} in this list, where $a_j = 1$. (For efficiency, after multiplying by each term, we reduce the result modulo m.) This gives us b^n. For example, to compute 3^{11} we first note that $11 = (1011)_2$, so that $3^{11} = 3^8 3^2 3^1$. By successively squaring, we find that $3^2 = 9$, $3^4 = 9^2 = 81$, and $3^8 = (81)^2 = 6561$. Consequently, $3^{11} = 3^8 3^2 3^1 = 6561 \cdot 9 \cdot 3 = 177{,}147$.

The algorithm successively finds b **mod** m, b^2 **mod** m, b^4 **mod** $m, \ldots, b^{2^{k-1}}$ **mod** m and multiplies together those terms b^{2^j} **mod** m where $a_j = 1$, finding the remainder of the product when divided by m after each multiplication. Pseudocode for this algorithm is shown in Algorithm 5. Note that in Algorithm 5 we can use the most efficient algorithm available to compute values of the **mod** function, not necessarily Algorithm 4.

Be sure to reduce modulo m after each multiplication!

ALGORITHM 5 Modular Exponentiation.

procedure *modular exponentiation*(b: integer, $n = (a_{k-1}a_{k-2}\ldots a_1a_0)_2$,
m: positive integers)
$x := 1$
$power := b$ **mod** m
for $i := 0$ **to** $k - 1$
  if $a_i = 1$ **then** $x := (x \cdot power)$ **mod** m
  $power := (power \cdot power)$ **mod** m
return x {x equals b^n **mod** m}

We illustrate how Algorithm 5 works in Example 12.

EXAMPLE 12 Use Algorithm 5 to find 3^{644} **mod** 645.

Solution: Algorithm 5 initially sets $x = 1$ and $power = 3$ **mod** $645 = 3$. In the computation of 3^{644} **mod** 645, this algorithm determines 3^{2^j} **mod** 645 for $j = 1, 2, \ldots, 9$ by successively squaring and reducing modulo 645. If $a_j = 1$ (where a_j is the bit in the jth position in the binary expansion of 644, which is $(1010000100)_2$), it multiplies the current value of x by 3^{2^j} **mod** 645 and reduces the result modulo 645. Here are the steps used:

$i = 0$: Because $a_0 = 0$, we have $x = 1$ and $power = 3^2$ **mod** $645 = 9$ **mod** $645 = 9$;
$i = 1$: Because $a_1 = 0$, we have $x = 1$ and $power = 9^2$ **mod** $645 = 81$ **mod** $645 = 81$;
$i = 2$: Because $a_2 = 1$, we have $x = 1 \cdot 81$ **mod** $645 = 81$ and $power = 81^2$ **mod** $645 = 6561$ **mod** $645 = 111$;
$i = 3$: Because $a_3 = 0$, we have $x = 81$ and $power = 111^2$ **mod** $645 = 12,321$ **mod** $645 = 66$;
$i = 4$: Because $a_4 = 0$, we have $x = 81$ and $power = 66^2$ **mod** $645 = 4356$ **mod** $645 = 486$;
$i = 5$: Because $a_5 = 0$, we have $x = 81$ and $power = 486^2$ **mod** $645 = 236,196$ **mod** $645 = 126$;
$i = 6$: Because $a_6 = 0$, we have $x = 81$ and $power = 126^2$ **mod** $645 = 15,876$ **mod** $645 = 396$;
$i = 7$: Because $a_7 = 1$, we find that $x = (81 \cdot 396)$ **mod** $645 = 471$ and $power = 396^2$ **mod** $645 = 156,816$ **mod** $645 = 81$;
$i = 8$: Because $a_8 = 0$, we have $x = 471$ and $power = 81^2$ **mod** $645 = 6561$ **mod** $645 = 111$;
$i = 9$: Because $a_9 = 1$, we find that $x = (471 \cdot 111)$ **mod** $645 = 36$.

This shows that following the steps of Algorithm 5 produces the result 3^{644} **mod** $645 = 36$. ◀

Algorithm 5 is quite efficient; it uses $O((\log m)^2 \log n)$ bit operations to find b^n **mod** m (see Exercise 58).

Exercises

1. Convert the decimal expansion of each of these integers to a binary expansion.
a) 231 **b)** 4532 **c)** 97644

2. Convert the decimal expansion of each of these integers to a binary expansion.
a) 321 **b)** 1023 **c)** 100632

3. Convert the binary expansion of each of these integers to a decimal expansion.
a) $(1\ 1111)_2$ **b)** $(10\ 0000\ 0001)_2$
c) $(1\ 0101\ 0101)_2$ **d)** $(110\ 1001\ 0001\ 0000)_2$

4. Convert the binary expansion of each of these integers to a decimal expansion.
a) $(1\ 1011)_2$ **b)** $(10\ 1011\ 0101)_2$
c) $(11\ 1011\ 1110)_2$ **d)** $(111\ 1100\ 0001\ 1111)_2$

5. Convert the octal expansion of each of these integers to a binary expansion.
a) $(572)_8$ **b)** $(1604)_8$
c) $(423)_8$ **d)** $(2417)_8$

6. Convert the binary expansion of each of these integers to an octal expansion.
a) $(1111\ 0111)_2$
b) $(1010\ 1010\ 1010)_2$
c) $(111\ 0111\ 0111\ 0111)_2$
d) $(101\ 0101\ 0101\ 0101)_2$

7. Convert the hexadecimal expansion of each of these integers to a binary expansion.
a) $(80E)_{16}$ **b)** $(135AB)_{16}$
c) $(ABBA)_{16}$ **d)** $(DEFACED)_{16}$

8. Convert $(BADFACED)_{16}$ from its hexadecimal expansion to its binary expansion.

9. Convert $(ABCDEF)_{16}$ from its hexadecimal expansion to its binary expansion.

10. Convert each of the integers in Exercise 6 from a binary expansion to a hexadecimal expansion.

11. Convert $(1011\ 0111\ 1011)_2$ from its binary expansion to its hexadecimal expansion.

12. Convert $(1\ 1000\ 0110\ 0011)_2$ from its binary expansion to its hexadecimal expansion.

13. Show that the hexadecimal expansion of a positive integer can be obtained from its binary expansion by grouping together blocks of four binary digits, adding initial zeros if necessary, and translating each block of four binary digits into a single hexadecimal digit.

14. Show that the binary expansion of a positive integer can be obtained from its hexadecimal expansion by translating each hexadecimal digit into a block of four binary digits.

15. Show that the octal expansion of a positive integer can be obtained from its binary expansion by grouping together blocks of three binary digits, adding initial zeros if necessary, and translating each block of three binary digits into a single octal digit.

16. Show that the binary expansion of a positive integer can be obtained from its octal expansion by translating each octal digit into a block of three binary digits.

17. Convert $(7345321)_8$ to its binary expansion and $(10\ 1011\ 1011)_2$ to its octal expansion.

18. Give a procedure for converting from the hexadecimal expansion of an integer to its octal expansion using binary notation as an intermediate step.

19. Give a procedure for converting from the octal expansion of an integer to its hexadecimal expansion using binary notation as an intermediate step.

20. Explain how to convert from binary to base 64 expansions and from base 64 expansions to binary expansions and from octal to base 64 expansions and from base 64 expansions to octal expansions.

21. Find the sum and the product of each of these pairs of numbers. Express your answers as a binary expansion.
a) $(100\ 0111)_2, (111\ 0111)_2$
b) $(1110\ 1111)_2, (1011\ 1101)_2$
c) $(10\ 1010\ 1010)_2, (1\ 1111\ 0000)_2$
d) $(10\ 0000\ 0001)_2, (11\ 1111\ 1111)_2$

22. Find the sum and product of each of these pairs of numbers. Express your answers as a base 3 expansion.
a) $(112)_3, (210)_3$
b) $(2112)_3, (12021)_3$
c) $(20001)_3, (1111)_3$
d) $(120021)_3, (2002)_3$

23. Find the sum and product of each of these pairs of numbers. Express your answers as an octal expansion.
a) $(763)_8, (147)_8$
b) $(6001)_8, (272)_8$
c) $(1111)_8, (777)_8$
d) $(54321)_8, (3456)_8$

24. Find the sum and product of each of these pairs of numbers. Express your answers as a hexadecimal expansion.
a) $(1AE)_{16}, (BBC)_{16}$
b) $(20CBA)_{16}, (A01)_{16}$
c) $(ABCDE)_{16}, (1111)_{16}$
d) $(E0000E)_{16}, (BAAA)_{16}$

25. Use Algorithm 5 to find 7^{644} **mod** 645.

26. Use Algorithm 5 to find 11^{644} **mod** 645.

27. Use Algorithm 5 to find 3^{2003} **mod** 99.

28. Use Algorithm 5 to find 123^{1001} **mod** 101.

29. Show that every positive integer can be represented uniquely as the sum of distinct powers of 2. [*Hint:* Consider binary expansions of integers.]

30. It can be shown that every integer can be uniquely represented in the form

$$e_k 3^k + e_{k-1} 3^{k-1} + \cdots + e_1 3 + e_0,$$

where $e_j = -1, 0,$ or 1 for $j = 0, 1, 2, \ldots, k$. Expansions of this type are called **balanced ternary expansions**. Find the balanced ternary expansions of

a) 5. **b)** 13. **c)** 37. **d)** 79.

31. Show that a positive integer is divisible by 3 if and only if the sum of its decimal digits is divisible by 3.

32. Show that a positive integer is divisible by 11 if and only if the difference of the sum of its decimal digits in even-numbered positions and the sum of its decimal digits in odd-numbered positions is divisible by 11.

33. Show that a positive integer is divisible by 3 if and only if the difference of the sum of its binary digits in even-numbered positions and the sum of its binary digits in odd-numbered positions is divisible by 3.

One's complement representations of integers are used to simplify computer arithmetic. To represent positive and negative integers with absolute value less than 2^{n-1}, a total of n bits is used. The leftmost bit is used to represent the sign. A 0 bit in this position is used for positive integers, and a 1 bit in this position is used for negative integers. For positive integers, the remaining bits are identical to the binary expansion of the integer. For negative integers, the remaining bits are obtained by first finding the binary expansion of the absolute value of the integer, and then taking the complement of each of these bits, where the complement of a 1 is a 0 and the complement of a 0 is a 1.

34. Find the one's complement representations, using bit strings of length six, of the following integers.

a) 22 **b)** 31 **c)** -7 **d)** -19

35. What integer does each of the following one's complement representations of length five represent?

a) 11001 **b)** 01101

c) 10001 **d)** 11111

36. If m is a positive integer less than 2^{n-1}, how is the one's complement representation of $-m$ obtained from the one's complement of m, when bit strings of length n are used?

37. How is the one's complement representation of the sum of two integers obtained from the one's complement representations of these integers?

38. How is the one's complement representation of the difference of two integers obtained from the one's complement representations of these integers?

39. Show that the integer m with one's complement representation $(a_{n-1}a_{n-2}\ldots a_1a_0)$ can be found using the equation $m = -a_{n-1}(2^{n-1} - 1) + a_{n-2}2^{n-2} + \cdots + a_1 \cdot 2 + a_0$.

Two's complement representations of integers are also used to simplify computer arithmetic and are used more commonly than one's complement representations. To represent an integer x with $-2^{n-1} \leq x \leq 2^{n-1} - 1$ for a specified positive integer n, a total of n bits is used. The leftmost bit is used to represent the sign. A 0 bit in this position is used for positive integers, and a 1 bit in this position is used for negative integers, just as in one's complement expansions. For a positive integer, the remaining bits are identical to the binary expansion of the integer. For a negative integer, the remaining bits are the bits of the binary expansion of $2^{n-1} - |x|$. Two's complement expansions of integers are often used by computers because addition and subtraction of integers can be performed easily using these expansions, where these integers can be either positive or negative.

40. Answer Exercise 34, but this time find the two's complement expansion using bit strings of length six.

41. Answer Exercise 35 if each expansion is a two's complement expansion of length five.

42. Answer Exercise 36 for two's complement expansions.

43. Answer Exercise 37 for two's complement expansions.

44. Answer Exercise 38 for two's complement expansions.

45. Show that the integer m with two's complement representation $(a_{n-1}a_{n-2}\ldots a_1a_0)$ can be found using the equation $m = -a_{n-1} \cdot 2^{n-1} + a_{n-2}2^{n-2} + \cdots + a_1 \cdot 2 + a_0$.

46. Give a simple algorithm for forming the two's complement representation of an integer from its one's complement representation.

47. Sometimes integers are encoded by using four-digit binary expansions to represent each decimal digit. This produces the **binary coded decimal** form of the integer. For instance, 791 is encoded in this way by 011110010001. How many bits are required to represent a number with n decimal digits using this type of encoding?

A **Cantor expansion** is a sum of the form

$$a_n n! + a_{n-1}(n-1)! + \cdots + a_2 2! + a_1 1!,$$

where a_i is an integer with $0 \leq a_i \leq i$ for $i = 1, 2, \ldots, n$.

48. Find the Cantor expansions of

a) 2. **b)** 7.
c) 19. **d)** 87.
e) 1000. **f)** 1,000,000.

***49.** Describe an algorithm that finds the Cantor expansion of an integer.

***50.** Describe an algorithm to add two integers from their Cantor expansions.

51. Add $(10111)_2$ and $(11010)_2$ by working through each step of the algorithm for addition given in the text.

52. Multiply $(1110)_2$ and $(1010)_2$ by working through each step of the algorithm for multiplication given in the text.

53. Describe an algorithm for finding the difference of two binary expansions.

54. Estimate the number of bit operations used to subtract two binary expansions.

55. Devise an algorithm that, given the binary expansions of the integers a and b, determines whether $a > b$, $a = b$, or $a < b$.

56. How many bit operations does the comparison algorithm from Exercise 55 use when the larger of a and b has n bits in its binary expansion?

57. Estimate the complexity of Algorithm 1 for finding the base b expansion of an integer n in terms of the number of divisions used.

***58.** Show that Algorithm 5 uses $O((\log m)^2 \log n)$ bit operations to find $b^n \bmod m$.

59. Show that Algorithm 4 uses $O(q \log a)$ bit operations, assuming that $a > d$.

4.3 Primes and Greatest Common Divisors

Introduction

In Section 4.1 we studied the concept of divisibility of integers. One important concept based on divisibility is that of a prime number. A prime is an integer greater than 1 that is divisible by no positive integers other than 1 and itself. The study of prime numbers goes back to ancient times. Thousands of years ago it was known that there are infinitely many primes; the proof of this fact, found in the works of Euclid, is famous for its elegance and beauty.

We will discuss the distribution of primes among the integers. We will describe some of the results about primes found by mathematicians in the last 400 years. In particular, we will introduce an important theorem, the fundamental theorem of arithmetic. This theorem, which asserts that every positive integer can be written uniquely as the product of primes in nondecreasing order, has many interesting consequences. We will also discuss some of the many old conjectures about primes that remain unsettled today.

Primes have become essential in modern cryptographic systems, and we will develop some of their properties important in cryptography. For example, finding large primes is essential in modern cryptography. The length of time required to factor large integers into their prime factors is the basis for the strength of some important modern cryptographic systems.

In this section we will also study the greatest common divisor of two integers, as well as the least common multiple of two integers. We will develop an important algorithm for computing greatest common divisors, called the Euclidean algorithm.

Primes

Every integer greater than 1 is divisible by at least two integers, because a positive integer is divisible by 1 and by itself. Positive integers that have exactly two different positive integer factors are called **primes**.

DEFINITION 1 An integer p greater than 1 is called *prime* if the only positive factors of p are 1 and p. A positive integer that is greater than 1 and is not prime is called *composite*.

Remark: The integer n is composite if and only if there exists an integer a such that $a \mid n$ and $1 < a < n$.

EXAMPLE 1 The integer 7 is prime because its only positive factors are 1 and 7, whereas the integer 9 is composite because it is divisible by 3. ◀

The primes are the building blocks of positive integers, as the fundamental theorem of arithmetic shows. The proof will be given in Section 5.2.

THEOREM 1 **THE FUNDAMENTAL THEOREM OF ARITHMETIC** Every integer greater than 1 can be written uniquely as a prime or as the product of two or more primes where the prime factors are written in order of nondecreasing size.

Example 2 gives some prime factorizations of integers.

EXAMPLE 2 The prime factorizations of 100, 641, 999, and 1024 are given by

Extra Examples

$$100 = 2 \cdot 2 \cdot 5 \cdot 5 = 2^2 5^2,$$
$$641 = 641,$$
$$999 = 3 \cdot 3 \cdot 3 \cdot 37 = 3^3 \cdot 37,$$
$$1024 = 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 \cdot 2 = 2^{10}.$$

◀

Trial Division

It is often important to show that a given integer is prime. For instance, in cryptology, large primes are used in some methods for making messages secret. One procedure for showing that an integer is prime is based on the following observation.

THEOREM 2 If n is a composite integer, then n has a prime divisor less than or equal to $\sqrt{n}$.

Proof: If n is composite, by the definition of a composite integer, we know that it has a factor a with $1 < a < n$. Hence, by the definition of a factor of a positive integer, we have $n = ab$, where b is a positive integer greater than 1. We will show that $a \leq \sqrt{n}$ or $b \leq \sqrt{n}$. If $a > \sqrt{n}$ and $b > \sqrt{n}$, then $ab > \sqrt{n} \cdot \sqrt{n} = n$, which is a contradiction. Consequently, $a \leq \sqrt{n}$ or $b \leq \sqrt{n}$. Because both a and b are divisors of n, we see that n has a positive divisor not exceeding $\sqrt{n}$. This divisor is either prime or, by the fundamental theorem of arithmetic, has a prime divisor less than itself. In either case, n has a prime divisor less than or equal to $\sqrt{n}$. ◁

From Theorem 2, it follows that an integer is prime if it is not divisible by any prime less than or equal to its square root. This leads to the brute-force algorithm known as **trial division**. To use trial division we divide n by all primes not exceeding $\sqrt{n}$ and conclude that n is prime if it is not divisible by any of these primes. In Example 3 we use trial division to show that 101 is prime.

EXAMPLE 3 Show that 101 is prime.

Solution: The only primes not exceeding $\sqrt{101}$ are 2, 3, 5, and 7. Because 101 is not divisible by 2, 3, 5, or 7 (the quotient of 101 and each of these integers is not an integer), it follows that 101 is prime. ◀

Because every integer has a prime factorization, it would be useful to have a procedure for finding this prime factorization. Consider the problem of finding the prime factorization of n. Begin by dividing n by successive primes, starting with the smallest prime, 2. If n has a prime factor, then by Theorem 3 a prime factor p not exceeding $\sqrt{n}$ will be found. So, if no prime

factor not exceeding $\sqrt{n}$ is found, then n is prime. Otherwise, if a prime factor p is found, continue by factoring n/p. Note that n/p has no prime factors less than p. Again, if n/p has no prime factor greater than or equal to p and not exceeding its square root, then it is prime. Otherwise, if it has a prime factor q, continue by factoring $n/(pq)$. This procedure is continued until the factorization has been reduced to a prime. This procedure is illustrated in Example 4.

EXAMPLE 4 Find the prime factorization of 7007.

Solution: To find the prime factorization of 7007, first perform divisions of 7007 by successive primes, beginning with 2. None of the primes 2, 3, and 5 divides 7007. However, 7 divides 7007, with $7007/7 = 1001$. Next, divide 1001 by successive primes, beginning with 7. It is immediately seen that 7 also divides 1001, because $1001/7 = 143$. Continue by dividing 143 by successive primes, beginning with 7. Although 7 does not divide 143, 11 does divide 143, and $143/11 = 13$. Because 13 is prime, the procedure is completed. It follows that $7007 = 7 \cdot 1001 = 7 \cdot 7 \cdot 143 = 7 \cdot 7 \cdot 11 \cdot 13$. Consequently, the prime factorization of 7007 is $7 \cdot 7 \cdot 11 \cdot 13 = 7^2 \cdot 11 \cdot 13$. ◀

Links Prime numbers were studied in ancient times for philosophical reasons. Today, there are highly practical reasons for their study. In particular, large primes play a crucial role in cryptography, as we will see in Section 4.6.

The Sieve of Eratosthenes

Note that composite integers not exceeding 100 must have a prime factor not exceeding 10. Because the only primes less than 10 are 2, 3, 5, and 7, the primes not exceeding 100 are these four primes and those positive integers greater than 1 and not exceeding 100 that are divisible by none of 2, 3, 5, or 7.

Links The **sieve of Eratosthenes** is used to find all primes not exceeding a specified positive integer. For instance, the following procedure is used to find the primes not exceeding 100. We begin with the list of all integers between 1 and 100. To begin the sieving process, the integers that are divisible by 2, other than 2, are deleted. Because 3 is the first integer greater than 2 that is left, all those integers divisible by 3, other than 3, are deleted. Because 5 is the next integer left after 3, those integers divisible by 5, other than 5, are deleted. The next integer left is 7, so those integers divisible by 7, other than 7, are deleted. Because all composite integers not exceeding 100 are divisible by 2, 3, 5, or 7, all remaining integers except 1 are prime. In Table 1, the panels display those integers deleted at each stage, where each integer divisible by 2, other than 2, is underlined in the first panel, each integer divisible by 3, other than 3, is underlined in the second panel, each integer divisible by 5, other than 5, is underlined in the third panel, and each integer divisible by 7, other than 7, is underlined in the fourth panel. The integers not underlined are the primes not exceeding 100. We conclude that the primes less than 100 are 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, and 97.

Links THE INFINITUDE OF PRIMES It has long been known that there are infinitely many primes. This means that whenever $p_1, p_2, \ldots, p_n$ are the n smallest primes, we know there is a larger

ERATOSTHENES (276 B.C.E.–194 B.C.E.) It is known that Eratosthenes was born in Cyrene, a Greek colony west of Egypt, and spent time studying at Plato's Academy in Athens. We also know that King Ptolemy II invited Eratosthenes to Alexandria to tutor his son and that later Eratosthenes became chief librarian at the famous library at Alexandria, a central repository of ancient wisdom. Eratosthenes was an extremely versatile scholar, writing on mathematics, geography, astronomy, history, philosophy, and literary criticism. Besides his work in mathematics, he is most noted for his chronology of ancient history and for his famous measurement of the size of the earth.

TABLE 1 The Sieve of Eratosthenes.

Integers divisible by 2 other than 2 receive an underline.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Integers divisible by 3 other than 3 receive an underline.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Integers divisible by 5 other than 5 receive an underline.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

Integers divisible by 7 other than 7 receive an underline; integers in color are prime.

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

prime not listed. We will prove this fact using a proof given by Euclid in his famous mathematics text, *The Elements*. This simple, yet elegant, proof is considered by many mathematicians to be among the most beautiful proofs in mathematics. It is the first proof presented in the book *Proofs from THE BOOK* [AiZi10], where THE BOOK refers to the imagined collection of perfect proofs that the famous mathematician Paul Erdős claimed is maintained by God. By the way, there are a vast number of different proofs than there are an infinitude of primes, and new ones are published surprisingly frequently.

THEOREM 3 There are infinitely many primes.

Proof: We will prove this theorem using a proof by contradiction. We assume that there are only finitely many primes, $p_1, p_2, \ldots, p_n$. Let

$$Q = p_1 p_2 \cdots p_n + 1.$$

By the fundamental theorem of arithmetic, Q is prime or else it can be written as the product of two or more primes. However, none of the primes p_j divides Q, for if $p_j \mid Q$, then p_j divides

$Q - p_1 p_2 \cdots p_n = 1$. Hence, there is a prime not in the list $p_1, p_2, \ldots, p_n$. This prime is either Q, if it is prime, or a prime factor of Q. This is a contradiction because we assumed that we have listed all the primes. Consequently, there are infinitely many primes. ◁

Remark: Note that in this proof we do *not* state that Q is prime! Furthermore, in this proof, we have given a nonconstructive existence proof that given any n primes, there is a prime not in this list. For this proof to be constructive, we would have had to explicitly give a prime not in our original list of n primes.

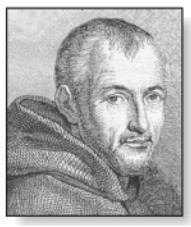
Because there are infinitely many primes, given any positive integer there are primes greater than this integer. There is an ongoing quest to discover larger and larger prime numbers; for almost all the last 300 years, the largest prime known has been an integer of the special form $2^p - 1$, where p is also prime. (Note that $2^n - 1$ cannot be prime when n is not prime; see Exercise 9.) Such primes are called **Mersenne primes**, after the French monk Marin Mersenne, who studied them in the seventeenth century. The reason that the largest known prime has usually been a Mersenne prime is that there is an extremely efficient test, known as the Lucas–Lehmer test, for determining whether $2^p - 1$ is prime. Furthermore, it is not currently possible to test numbers not of this or certain other special forms anywhere near as quickly to determine whether they are prime.

EXAMPLE 5 The numbers $2^2 - 1 = 3$, $2^3 - 1 = 7$, $2^5 - 1 = 31$ and $2^7 - 1 = 127$ are Mersenne primes, while $2^{11} - 1 = 2047$ is not a Mersenne prime because $2047 = 23 \cdot 89$. ◀

Links Progress in finding Mersenne primes has been steady since computers were invented. As of early 2011, 47 Mersenne primes were known, with 16 found since 1990. The largest Mersenne prime known (again as of early 2011) is $2^{43,112,609} - 1$, a number with nearly 13 million decimal digits, which was shown to be prime in 2008. A communal effort, the Great Internet Mersenne Prime Search (GIMPS), is devoted to the search for new Mersenne primes. You can join this search, and if you are lucky, find a new Mersenne prime and possibly even win a cash prize. By the way, even the search for Mersenne primes has practical implications. One quality control test for supercomputers has been to replicate the Lucas–Lehmer test that establishes the primality of a large Mersenne prime. (See [Ro10] for more information about the quest for finding Mersenne primes.)

THE DISTRIBUTION OF PRIMES Theorem 3 tells us that there are infinitely many primes. However, how many primes are less than a positive number x? This question interested mathematicians for many years; in the late eighteenth century, mathematicians produced large tables

Links



MARIN MERSENNE (1588–1648) Mersenne was born in Maine, France, into a family of laborers and attended the College of Mans and the Jesuit College at La Flèche. He continued his education at the Sorbonne, studying theology from 1609 to 1611. He joined the religious order of the Minims in 1611, a group whose name comes from the word *minimi* (the members of this group were extremely humble; they considered themselves the least of all religious orders). Besides prayer, the members of this group devoted their energy to scholarship and study. In 1612 he became a priest at the Place Royale in Paris; between 1614 and 1618 he taught philosophy at the Minim Convent at Nevers. He returned to Paris in 1619, where his cell in the Minims de l'Annociade became a place for meetings of French scientists, philosophers, and mathematicians, including Fermat and Pascal. Mersenne corresponded extensively with scholars throughout Europe, serving as a clearinghouse for mathematical and scientific knowledge, a function later served by mathematical journals (and today also by the Internet). Mersenne wrote books covering mechanics, mathematical physics, mathematics, music, and acoustics. He studied prime numbers and tried unsuccessfully to construct a formula representing all primes. In 1644 Mersenne claimed that $2^p - 1$ is prime for $p = 2, 3, 5, 7, 13, 17, 19, 31, 67, 127, 257$ but is composite for all other primes less than 257. It took over 300 years to determine that Mersenne's claim was wrong five times. Specifically, $2^p - 1$ is not prime for $p = 67$ and $p = 257$ but is prime for $p = 61$, $p = 87$, and $p = 107$. It is also noteworthy that Mersenne defended two of the most famous men of his time, Descartes and Galileo, from religious critics. He also helped expose alchemists and astrologers as frauds.

of prime numbers to gather evidence concerning the distribution of primes. Using this evidence, the great mathematicians of the day, including Gauss and Legendre, conjectured, but did not prove, Theorem 4.

THEOREM 4 **THE PRIME NUMBER THEOREM** The ratio of the number of primes not exceeding x and $x/\ln x$ approaches 1 as x grows without bound. (Here $\ln x$ is the natural logarithm of x.)

Links The prime number theorem was first proved in 1896 by the French mathematician Jacques Hadamard and the Belgian mathematician Charles-Jean-Gustave-Nicholas de la Vallée-Poussin using the theory of complex variables. Although proofs not using complex variables have been found, all known proofs of the prime number theorem are quite complicated.

We can use the prime number theorem to estimate the odds that a randomly chosen number is prime. The prime number theorem tells us that the number of primes not exceeding x can be approximated by $x/\ln x$. Consequently, the odds that a randomly selected positive integer less than n is prime are approximately $(n/\ln n)/n = 1/\ln n$. Sometimes we need to find a prime with a particular number of digits. We would like an estimate of how many integers with a particular number of digits we need to select before we encounter a prime. Using the prime number theorem and calculus, it can be shown that the probability that an integer n is prime is also approximately $1/\ln n$. For example, the odds that an integer near 10^{1000} is prime are approximately $1/\ln 10^{1000}$, which is approximately $1/2300$. (Of course, by choosing only odd numbers, we double our chances of finding a prime.)

Using trial division with Theorem 2 gives procedures for factoring and for primality testing. However, these procedures are not efficient algorithms; many much more practical and efficient algorithms for these tasks have been developed. Factoring and primality testing have become important in the applications of number theory to cryptography. This has led to a great interest in developing efficient algorithms for both tasks. Clever procedures have been devised in the last 30 years for efficiently generating large primes. Moreover, in 2002, an important theoretical discovery was made by Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. They showed there is a polynomial-time algorithm in the number of bits in the binary expansion of an integer for determining whether a positive integer is prime. Algorithms based on their work use $O((\log n)^6)$ bit operations to determine whether a positive integer n is prime.

However, even though powerful new factorization methods have been developed in the same time frame, factoring large numbers remains extraordinarily more time-consuming than primality testing. No polynomial-time algorithm for factoring integers is known. Nevertheless, the challenge of factoring large numbers interests many people. There is a communal effort on the Internet to factor large numbers, especially those of the special form $k^n \pm 1$, where k is a small positive integer and n is a large positive integer (such numbers are called *Cunningham numbers*). At any given time, there is a list of the "Ten Most Wanted" large numbers of this type awaiting factorization.

PRIMES AND ARITHMETIC PROGRESSIONS Every odd integer is in one of the two arithmetic progressions $4k + 1$ or $4k + 3$, $k = 1, 2, \ldots$. Because we know that there are infinitely many primes, we can ask whether there are infinitely many primes in both of these arithmetic progressions. The primes 5, 13, 17, 29, 37, 41, ... are in the arithmetic progression $4k + 1$; the primes 3, 7, 11, 19, 23, 31, 43, ... are in the arithmetic progression $4k + 3$. Looking at the evidence hints that there may be infinitely many primes in both progressions. What about other arithmetic progressions $ak + b$, $k = 1, 2, \ldots$, where no integer greater than one divides both a and b? Do they contain infinitely many primes? The answer was provided by the German mathematician G. Lejeune Dirichlet, who proved that every such arithmetic progression contains infinitely many primes. His proof, and all proofs found later, are beyond the scope of this book.

However, it is possible to prove special cases of Dirichlet's theorem using the ideas developed in this book. For example, Exercises 54 and 55 ask for proofs that there are infinitely many primes in the arithmetic progressions $3k + 2$ and $4k + 3$, where k is a positive integer. (The hint for each of these exercises supplies the basic idea needed for the proof.)

We have explained that every arithmetic progression $ak + b$, $k = 1, 2, \ldots$, where a and b have no common factor greater than one, contains infinitely many primes. But are there long arithmetic progressions made up of just primes? For example, some exploration shows that 5, 11, 17, 23, 29 is an arithmetic progression of five primes and 199, 409, 619, 829, 1039, 1249, 1459, 1669, 1879, 2089 is an arithmetic progression of ten primes. In the 1930s, the famous mathematician Paul Erdős conjectured that for every positive integer n greater than two, there is an arithmetic progression of length n made up entirely of primes. In 2006, Ben Green and Terence Tao were able to prove this conjecture. Their proof, considered to be a mathematical tour de force, is a nonconstructive proof that combines powerful ideas from several advanced areas of mathematics.

Conjectures and Open Problems About Primes

Number theory is noted as a subject for which it is easy to formulate conjectures, some of which are difficult to prove and others that remained open problems for many years. We will describe some conjectures in number theory and discuss their status in Examples 6–9.

EXAMPLE 6 Extra Examples

It would be useful to have a function $f(n)$ such that $f(n)$ is prime for all positive integers n. If we had such a function, we could find large primes for use in cryptography and other applications. Looking for such a function, we might check out different polynomial functions, as some mathematicians did several hundred years ago. After a lot of computation we may encounter the polynomial $f(n) = n^2 - n + 41$. This polynomial has the interesting property that $f(n)$ is prime for all positive integers n not exceeding 40. [We have $f(1) = 41$, $f(2) = 43$, $f(3) = 47$, $f(4) = 53$, and so on.] This can lead us to the conjecture that $f(n)$ is prime for all positive integers n. Can we settle this conjecture?

Solution: Perhaps not surprisingly, this conjecture turns out to be false; we do not have to look far to find a positive integer n for which $f(n)$ is composite, because $f(41) = 41^2 - 41 + 41 = 41^2$. Because $f(n) = n^2 - n + 41$ is prime for all positive integers n with $1 \leq n \leq 40$, we might

TERENCE TAO (BORN 1975) Tao was born in Australia. His father is a pediatrician and his mother taught mathematics at a Hong Kong secondary school. Tao was a child prodigy, teaching himself arithmetic at the age of two. At 10, he became the youngest contestant at the International Mathematical Olympiad (IMO); he won an IMO gold medal at 13. Tao received his bachelors and masters degrees when he was 17, and began graduate studies at Princeton, receiving his Ph.D. in three years. In 1996 he became a faculty member at UCLA, where he continues to work.

Tao is extremely versatile; he enjoys working on problems in diverse areas, including harmonic analysis, partial differential equations, number theory, and combinatorics. You can follow his work by reading his blog where he discusses progress on various problems. His most famous result is the Green-Tao theorem, which says that there are arbitrarily long arithmetic progressions of primes. Tao has made important contributions to the applications of mathematics, such as developing a method for reconstructing digital images using the least possible amount of information.
Tao has an amazing reputation among mathematicians; he has become a Mr. Fix-It for researchers in mathematics. The well-known mathematician Charles Fefferman, himself a child prodigy, has said that "if you're stuck on a problem, then one way out is to interest Terence Tao." In 2006 Tao was awarded a Fields Medal, the most prestigious award for mathematicians under the age of 40. He was also awarded a MacArthur Fellowship in 2006, and in 2008, he received the Allan T. Waterman award, which came with a $500,000 cash prize to support research work of scientists early in their career. Tao's wife Laura is an engineer at the Jet Propulsion Laboratory.

be tempted to find a different polynomial with the property that $f(n)$ is prime for *all* positive integers n. However, there is no such polynomial. It can be shown that for every polynomial $f(n)$ with integer coefficients, there is a positive integer y such that $f(y)$ is composite. (See Exercise 23 in the Supplementary Exercises.) ◀

Many famous problems about primes still await ultimate resolution by clever people. We describe a few of the most accessible and better known of these open problems in Examples 7–9. Number theory is noted for its wealth of easy-to-understand conjectures that resist attack by all but the most sophisticated techniques, or simply resist all attacks. We present these conjectures to show that many questions that seem relatively simple remain unsettled even in the twenty-first century.

EXAMPLE 7 **Goldbach's Conjecture** In 1742, Christian Goldbach, in a letter to Leonhard Euler, conjectured that every odd integer $n, n > 5$, is the sum of three primes. Euler replied that this conjecture is equivalent to the conjecture that every even integer n, $n > 2$, is the sum of two primes (see Exercise 21 in the Supplementary Exercises). The conjecture that every even integer $n, n > 2$, is the sum of two primes is now called **Goldbach's conjecture**. We can check this conjecture for small even numbers. For example, $4 = 2 + 2$, $6 = 3 + 3$, $8 = 5 + 3$, $10 = 7 + 3$, $12 = 7 + 5$, and so on. Goldbach's conjecture was verified by hand calculations for numbers up to the millions prior to the advent of computers. With computers it can be checked for extremely large numbers. As of mid 2011, the conjecture has been checked for all positive even integers up to $1.6 \cdot 10^{18}$.

Links

Although no proof of Goldbach's conjecture has been found, most mathematicians believe it is true. Several theorems have been proved, using complicated methods from analytic number theory far beyond the scope of this book, establishing results weaker than Goldbach's conjecture. Among these are the result that every even integer greater than 2 is the sum of at most six primes (proved in 1995 by O. Ramaré) and that every sufficiently large positive integer is the sum of a prime and a number that is either prime or the product of two primes (proved in 1966 by J. R. Chen). Perhaps Goldbach's conjecture will be settled in the not too distant future. ◀

EXAMPLE 8 There are many conjectures asserting that there are infinitely many primes of certain special forms. A conjecture of this sort is the conjecture that there are infinitely many primes of the form $n^2 + 1$, where n is a positive integer. For example, $5 = 2^2 + 1$, $17 = 4^2 + 1$, $37 = 6^2 + 1$, and so on. The best result currently known is that there are infinitely many positive integers n such that $n^2 + 1$ is prime or the product of at most two primes (proved by Henryk Iwaniec in 1973 using advanced techniques from analytic number theory, far beyond the scope of this book). ◀

Links

EXAMPLE 9 **The Twin Prime Conjecture** **Twin primes** are pairs of primes that differ by 2, such as 3 and 5, 5 and 7, 11 and 13, 17 and 19, and 4967 and 4969. The twin prime conjecture asserts that there are infinitely many twin primes. The strongest result proved concerning twin primes is that there are infinitely many pairs p and $p + 2$, where p is prime and $p + 2$ is prime or the product of two primes (proved by J. R. Chen in 1966). The world's record for twin primes, as of mid 2011, consists of the numbers $65{,}516{,}468{,}355 \cdot 2^{333{,}333} \pm 1$, which have 100,355 decimal digits. ◀

Links

Links

CHRISTIAN GOLDBACH (1690–1764) Christian Goldbach was born in Königsberg, Prussia, the city noted for its famous bridge problem (which will be studied in Section 10.5). He became professor of mathematics at the Academy in St. Petersburg in 1725. In 1728 Goldbach went to Moscow to tutor the son of the Tsar. He entered the world of politics when, in 1742, he became a staff member in the Russian Ministry of Foreign Affairs. Goldbach is best known for his correspondence with eminent mathematicians, including Euler and Bernoulli, for his famous conjectures in number theory, and for several contributions to analysis.

Greatest Common Divisors and Least Common Multiples

The largest integer that divides both of two integers is called the **greatest common divisor** of these integers.

DEFINITION 2 Let a and b be integers, not both zero. The largest integer d such that $d \mid a$ and $d \mid b$ is called the *greatest common divisor* of a and b. The greatest common divisor of a and b is denoted by $\gcd(a, b)$.

The greatest common divisor of two integers, not both zero, exists because the set of common divisors of these integers is nonempty and finite. One way to find the greatest common divisor of two integers is to find all the positive common divisors of both integers and then take the largest divisor. This is done in Examples 10 and 11. Later, a more efficient method of finding greatest common divisors will be given.

EXAMPLE 10 What is the greatest common divisor of 24 and 36?

Solution: The positive common divisors of 24 and 36 are 1, 2, 3, 4, 6, and 12. Hence, $\gcd(24, 36) = 12$. ◀

EXAMPLE 11 What is the greatest common divisor of 17 and 22?

Solution: The integers 17 and 22 have no positive common divisors other than 1, so that $\gcd(17, 22) = 1$. ◀

Because it is often important to specify that two integers have no common positive divisor other than 1, we have Definition 3.

DEFINITION 3 The integers a and b are *relatively prime* if their greatest common divisor is 1.

EXAMPLE 12 By Example 11 it follows that the integers 17 and 22 are relatively prime, because $\gcd(17, 22) = 1$. ◀

Because we often need to specify that no two integers in a set of integers have a common positive divisor greater than 1, we make Definition 4.

DEFINITION 4 The integers $a_1, a_2, \ldots, a_n$ are *pairwise relatively prime* if $\gcd(a_i, a_j) = 1$ whenever $1 \leq i < j \leq n$.

EXAMPLE 13 Determine whether the integers 10, 17, and 21 are pairwise relatively prime and whether the integers 10, 19, and 24 are pairwise relatively prime.

Solution: Because $\gcd(10, 17) = 1$, $\gcd(10, 21) = 1$, and $\gcd(17, 21) = 1$, we conclude that 10, 17, and 21 are pairwise relatively prime.

Because $\gcd(10, 24) = 2 > 1$, we see that 10, 19, and 24 are not pairwise relatively prime. ◀

Another way to find the greatest common divisor of two positive integers is to use the prime factorizations of these integers. Suppose that the prime factorizations of the positive integers a and b are

$$a = p_1^{a_1} p_2^{a_2} \cdots p_n^{a_n}, \; b = p_1^{b_1} p_2^{b_2} \cdots p_n^{b_n},$$

where each exponent is a nonnegative integer, and where all primes occurring in the prime factorization of either a or b are included in both factorizations, with zero exponents if necessary. Then $\gcd(a, b)$ is given by

$$\gcd(a, b) = p_1^{\min(a_1, b_1)} p_2^{\min(a_2, b_2)} \cdots p_n^{\min(a_n, b_n)},$$

where $\min(x, y)$ represents the minimum of the two numbers x and y. To show that this formula for $\gcd(a, b)$ is valid, we must show that the integer on the right-hand side divides both a and b, and that no larger integer also does. This integer does divide both a and b, because the power of each prime in the factorization does not exceed the power of this prime in either the factorization of a or that of b. Further, no larger integer can divide both a and b, because the exponents of the primes in this factorization cannot be increased, and no other primes can be included.

EXAMPLE 14 Because the prime factorizations of 120 and 500 are $120 = 2^3 \cdot 3 \cdot 5$ and $500 = 2^2 \cdot 5^3$, the greatest common divisor is

$$\gcd(120, 500) = 2^{\min(3, 2)} 3^{\min(1, 0)} 5^{\min(1, 3)} = 2^2 3^0 5^1 = 20. \quad \blacktriangleleft$$

Prime factorizations can also be used to find the **least common multiple** of two integers.

DEFINITION 5 The *least common multiple* of the positive integers a and b is the smallest positive integer that is divisible by both a and b. The least common multiple of a and b is denoted by $\operatorname{lcm}(a, b)$.

The least common multiple exists because the set of integers divisible by both a and b is nonempty (as ab belongs to this set, for instance), and every nonempty set of positive integers has a least element (by the well-ordering property, which will be discussed in Section 5.2). Suppose that the prime factorizations of a and b are as before. Then the least common multiple of a and b is given by

$$\operatorname{lcm}(a, b) = p_1^{\max(a_1, b_1)} p_2^{\max(a_2, b_2)} \cdots p_n^{\max(a_n, b_n)},$$

where $\max(x, y)$ denotes the maximum of the two numbers x and y. This formula is valid because a common multiple of a and b has at least $\max(a_i, b_i)$ factors of p_i in its prime factorization, and the least common multiple has no other prime factors besides those in a and b.

EXAMPLE 15 What is the least common multiple of $2^3 3^5 7^2$ and $2^4 3^3$?

Solution: We have

$$\operatorname{lcm}(2^3 3^5 7^2, 2^4 3^3) = 2^{\max(3, 4)} 3^{\max(5, 3)} 7^{\max(2, 0)} = 2^4 3^5 7^2. \quad \blacktriangleleft$$

Theorem 5 gives the relationship between the greatest common divisor and least common multiple of two integers. It can be proved using the formulae we have derived for these quantities. The proof of this theorem is left as Exercise 31.

THEOREM 5 Let a and b be positive integers. Then

$$ab = \gcd(a, b) \cdot \operatorname{lcm}(a, b).$$

The Euclidean Algorithm

Links

Computing the greatest common divisor of two integers directly from the prime factorizations of these integers is inefficient. The reason is that it is time-consuming to find prime factorizations. We will give a more efficient method of finding the greatest common divisor, called the **Euclidean algorithm**. This algorithm has been known since ancient times. It is named after the ancient Greek mathematician Euclid, who included a description of this algorithm in his book *The Elements*.

Before describing the Euclidean algorithm, we will show how it is used to find gcd(91, 287). First, divide 287, the larger of the two integers, by 91, the smaller, to obtain

$$287 = 91 \cdot 3 + 14.$$

Any divisor of 91 and 287 must also be a divisor of $287 - 91 \cdot 3 = 14$. Also, any divisor of 91 and 14 must also be a divisor of $287 = 91 \cdot 3 + 14$. Hence, the greatest common divisor of 91 and 287 is the same as the greatest common divisor of 91 and 14. This means that the problem of finding gcd(91, 287) has been reduced to the problem of finding gcd(91, 14).

Next, divide 91 by 14 to obtain

$$91 = 14 \cdot 6 + 7.$$

Because any common divisor of 91 and 14 also divides $91 - 14 \cdot 6 = 7$ and any common divisor of 14 and 7 divides 91, it follows that gcd(91, 14) = gcd(14, 7).

Continue by dividing 14 by 7, to obtain

$$14 = 7 \cdot 2.$$

Because 7 divides 14, it follows that gcd(14, 7) = 7. Furthermore, because gcd(287, 91) = gcd(91, 14) = gcd(14, 7) = 7, the original problem has been solved.

We now describe how the Euclidean algorithm works in generality. We will use successive divisions to reduce the problem of finding the greatest common divisor of two positive integers to the same problem with smaller integers, until one of the integers is zero.

The Euclidean algorithm is based on the following result about greatest common divisors and the division algorithm.

Links

EUCLID (325 B.C.E.–265 B.C.E.) Euclid was the author of the most successful mathematics book ever written, *The Elements*, which appeared in over 1000 different editions from ancient to modern times. Little is known about Euclid's life, other than that he taught at the famous academy at Alexandria in Egypt. Apparently, Euclid did not stress applications. When a student asked what he would get by learning geometry, Euclid explained that knowledge was worth acquiring for its own sake and told his servant to give the student a coin "because he must make a profit from what he learns."

LEMMA 1 Let $a = bq + r$, where a, b, q, and r are integers. Then $\gcd(a, b) = \gcd(b, r)$.

Proof: If we can show that the common divisors of a and b are the same as the common divisors of b and r, we will have shown that $\gcd(a, b) = \gcd(b, r)$, because both pairs must have the same *greatest* common divisor.

So suppose that d divides both a and b. Then it follows that d also divides $a - bq = r$ (from Theorem 1 of Section 4.1). Hence, any common divisor of a and b is also a common divisor of b and r.

Likewise, suppose that d divides both b and r. Then d also divides $bq + r = a$. Hence, any common divisor of b and r is also a common divisor of a and b.

Consequently, $\gcd(a, b) = \gcd(b, r)$. ◁

Suppose that a and b are positive integers with $a \geq b$. Let $r_0 = a$ and $r_1 = b$. When we successively apply the division algorithm, we obtain

$$\begin{aligned}
r_0 &= r_1 q_1 + r_2 && 0 \leq r_2 < r_1, \\
r_1 &= r_2 q_2 + r_3 && 0 \leq r_3 < r_2, \\
&\;\;\vdots \\
r_{n-2} &= r_{n-1} q_{n-1} + r_n && 0 \leq r_n < r_{n-1}, \\
r_{n-1} &= r_n q_n.
\end{aligned}$$

Eventually a remainder of zero occurs in this sequence of successive divisions, because the sequence of remainders $a = r_0 > r_1 > r_2 > \cdots \geq 0$ cannot contain more than a terms. Furthermore, it follows from Lemma 1 that

$$\begin{aligned}
\gcd(a, b) &= \gcd(r_0, r_1) = \gcd(r_1, r_2) = \cdots = \gcd(r_{n-2}, r_{n-1}) \\
&= \gcd(r_{n-1}, r_n) = \gcd(r_n, 0) = r_n.
\end{aligned}$$

Hence, the greatest common divisor is the last nonzero remainder in the sequence of divisions.

EXAMPLE 16 Find the greatest common divisor of 414 and 662 using the Euclidean algorithm.

Solution: Successive uses of the division algorithm give:

$$\begin{aligned}
662 &= 414 \cdot 1 + 248 \\
414 &= 248 \cdot 1 + 166 \\
248 &= 166 \cdot 1 + 82 \\
166 &= 82 \cdot 2 + 2 \\
82 &= 2 \cdot 41.
\end{aligned}$$

Hence, $\gcd(414, 662) = 2$, because 2 is the last nonzero remainder. ◁

The Euclidean algorithm is expressed in pseudocode in Algorithm 1.

ALGORITHM 1 **The Euclidean Algorithm.**

procedure $gcd(a, b$: positive integers)
$x := a$
$y := b$
while $y \neq 0$
    $r := x$ **mod** y
    $x := y$
    $y := r$
return $x\{\gcd(a, b)$ is $x\}$

In Algorithm 1, the initial values of x and y are a and b, respectively. At each stage of the procedure, x is replaced by y, and y is replaced by x **mod** y, which is the remainder when x is divided by y. This process is repeated as long as $y \neq 0$. The algorithm terminates when $y = 0$, and the value of x at that point, the last nonzero remainder in the procedure, is the greatest common divisor of a and b.

We will study the time complexity of the Euclidean algorithm in Section 5.3, where we will show that the number of divisions required to find the greatest common divisor of a and b, where $a \geq b$, is $O(\log b)$.

gcds as Linear Combinations

An important result we will use throughout the remainder of this section is that the greatest common divisor of two integers a and b can be expressed in the form

$$sa + tb,$$

where s and t are integers. In other words, $\gcd(a, b)$ can be expressed as a **linear combination** with integer coefficients of a and b. For example, $\gcd(6, 14) = 2$, and $2 = (-2) \cdot 6 + 1 \cdot 14$. We state this fact as Theorem 6.

THEOREM 6 **BÉZOUT'S THEOREM** If a and b are positive integers, then there exist integers s and t such that $\gcd(a, b) = sa + tb$.

Links

ÉTIENNE BÉZOUT (1730–1783) Bézout was born in Nemours, France, where his father was a magistrate. Reading the writings of the great mathematician Leonhard Euler enticed him to become a mathematician. In 1758 he was appointed to a position at the Académie des Sciences in Paris; in 1763 he was appointed examiner of the Gardes de la Marine, where he was assigned the task of writing mathematics textbooks. This assignment led to a four-volume textbook completed in 1767. Bézout is well known for his six-volume comprehensive textbook on mathematics. His textbooks were extremely popular and were studied by many generations of students hoping to enter the École Polytechnique, the famous engineering and science school. His books were translated into English and used in North America, including at Harvard.

His most important original work was published in 1779 in the book *Théorie générale des équations algébriques*, where he introduced important methods for solving simultaneous polynomial equations in many unknowns. The most well-known result in this book is now called *Bézout's theorem*, which in its general form tells us that the number of common points on two plane algebraic curves equals the product of the degrees of these curves. Bézout is also credited with inventing the determinant (which was called the Bézoutian by the great English mathematician James Joseph Sylvester). He was considered to be a kind person with a warm heart, although he had a reserved and somber personality. He was happily married and a father.

DEFINITION 6

If a and b are positive integers, then integers s and t such that $\gcd(a, b) = sa + tb$ are called *Bézout coefficients* of a and b (after Étienne Bézout, a French mathematician of the eighteenth century). Also, the equation $\gcd(a, b) = sa + tb$ is called *Bézout's identity*.

We will not give a formal proof of Theorem 6 here (see Exercise 36 in Section 5.2 and [Ro10] for proofs). We will provide an example of a general method that can be used to find a linear combination of two integers equal to their greatest common divisor. (In this section, we will assume that a linear combination has integer coefficients.) The method proceeds by working backward through the divisions of the Euclidean algorithm, so this method requires a forward pass and a backward pass through the steps of the Euclidean algorithm. (In the exercises we will describe an algorithm called the **extended Euclidean algorithm**, which can be used to express $\gcd(a, b)$ as a linear combination of a and b using a single pass through the steps of the Euclidean algorithm; see the preamble to Exercise 41.)

EXAMPLE 17 Express $\gcd(252, 198) = 18$ as a linear combination of 252 and 198.

Solution: To show that $\gcd(252, 198) = 18$, the Euclidean algorithm uses these divisions:

$$\begin{aligned} 252 &= 1 \cdot 198 + 54 \\ 198 &= 3 \cdot 54 + 36 \\ 54 &= 1 \cdot 36 + 18 \\ 36 &= 2 \cdot 18. \end{aligned}$$

Using the next-to-last division (the third division), we can express $\gcd(252, 198) = 18$ as a linear combination of 54 and 36. We find that

$$18 = 54 - 1 \cdot 36.$$

The second division tells us that

$$36 = 198 - 3 \cdot 54.$$

Substituting this expression for 36 into the previous equation, we can express 18 as a linear combination of 54 and 198. We have

$$18 = 54 - 1 \cdot 36 = 54 - 1 \cdot (198 - 3 \cdot 54) = 4 \cdot 54 - 1 \cdot 198.$$

The first division tells us that

$$54 = 252 - 1 \cdot 198.$$

Substituting this expression for 54 into the previous equation, we can express 18 as a linear combination of 252 and 198. We conclude that

$$18 = 4 \cdot (252 - 1 \cdot 198) - 1 \cdot 198 = 4 \cdot 252 - 5 \cdot 198,$$

completing the solution. ◀

We will use Theorem 6 to develop several useful results. One of our goals will be to prove the part of the fundamental theorem of arithmetic asserting that a positive integer has at most one prime factorization. We will show that if a positive integer has a factorization into primes, where the primes are written in nondecreasing order, then this factorization is unique.

First, we need to develop some results about divisibility.

LEMMA 2 If a, b, and c are positive integers such that $\gcd(a, b) = 1$ and $a \mid bc$, then $a \mid c$.

Proof: Because $\gcd(a, b) = 1$, by Bézout's theorem there are integers s and t such that

$$sa + tb = 1.$$

Multiplying both sides of this equation by c, we obtain

$$sac + tbc = c.$$

We can now use Theorem 1 of Section 4.1 to show that $a \mid c$. By part (ii) of that theorem, $a \mid tbc$. Because $a \mid sac$ and $a \mid tbc$, by part (i) of that theorem, we conclude that a divides $sac + tbc$. Because $sac + tbc = c$, we conclude that $a \mid c$, completing the proof. ◁

We will use the following generalization of Lemma 2 in the proof of uniqueness of prime factorizations. (The proof of Lemma 3 is left as Exercise 64 in Section 5.1, because it can be most easily carried out using the method of mathematical induction, covered in that section.)

LEMMA 3 If p is a prime and $p \mid a_1 a_2 \cdots a_n$, where each a_i is an integer, then $p \mid a_i$ for some i.

We can now show that a factorization of an integer into primes is unique. That is, we will show that every integer can be written as the product of primes in nondecreasing order in at most one way. This is part of the fundamental theorem of arithmetic. We will prove the other part, that every integer has a factorization into primes, in Section 5.2.

Proof (of the uniqueness of the prime factorization of a positive integer): We will use a proof by contradiction. Suppose that the positive integer n can be written as the product of primes in two different ways, say, $n = p_1 p_2 \cdots p_s$ and $n = q_1 q_2 \cdots q_t$, each p_i and q_j are primes such that $p_1 \le p_2 \le \cdots \le p_s$ and $q_1 \le q_2 \le \cdots \le q_t$.

When we remove all common primes from the two factorizations, we have

$$p_{i_1} p_{i_2} \cdots p_{i_u} = q_{j_1} q_{j_2} \cdots q_{j_v},$$

where no prime occurs on both sides of this equation and u and v are positive integers. By Lemma 3 it follows that p_{i_1} divides q_{j_k} for some k. Because no prime divides another prime, this is impossible. Consequently, there can be at most one factorization of n into primes in nondecreasing order. ◁

Lemma 2 can also be used to prove a result about dividing both sides of a congruence by the same integer. We have shown (Theorem 5 in Section 4.1) that we can multiply both sides of a congruence by the same integer. However, dividing both sides of a congruence by an integer does not always produce a valid congruence, as Example 18 shows.

EXAMPLE 18 The congruence $14 \equiv 8 \pmod 6$ holds, but both sides of this congruence cannot be divided by 2 to produce a valid congruence because $14/2 = 7$ and $8/2 = 4$, but $7 \not\equiv 4 \pmod 6$. ◀

Although we cannot divide both sides of a congruence by any integer to produce a valid congruence, we can if this integer is relatively prime to the modulus. Theorem 7 establishes this important fact. We use Lemma 2 in the proof.

THEOREM 7 Let m be a positive integer and let a, b, and c be integers. If $ac \equiv bc \pmod{m}$ and $\gcd(c, m) = 1$, then $a \equiv b \pmod{m}$.

Proof: Because $ac \equiv bc \pmod{m}$, $m \mid ac - bc = c(a - b)$. By Lemma 2, because $\gcd(c, m) = 1$, it follows that $m \mid a - b$. We conclude that $a \equiv b \pmod{m}$. ◁

Exercises

1. Determine whether each of these integers is prime.
a) 21 **b)** 29
c) 71 **d)** 97
e) 111 **f)** 143

2. Determine whether each of these integers is prime.
a) 19 **b)** 27
c) 93 **d)** 101
e) 107 **f)** 113

3. Find the prime factorization of each of these integers.
a) 88 **b)** 126 **c)** 729
d) 1001 **e)** 1111 **f)** 909,090

4. Find the prime factorization of each of these integers.
a) 39 **b)** 81 **c)** 101
d) 143 **e)** 289 **f)** 899

5. Find the prime factorization of 10!.

***6.** How many zeros are there at the end of 100!?

7. Express in pseudocode the trial division algorithm for determining whether an integer is prime.

8. Express in pseudocode the algorithm described in the text for finding the prime factorization of an integer.

9. Show that if $a^m + 1$ is composite if a and m are integers greater than 1 and m is odd. [*Hint:* Show that $x + 1$ is a factor of the polynomial $x^m + 1$ if m is odd.]

10. Show that if $2^m + 1$ is an odd prime, then $m = 2^n$ for some nonnegative integer n. [*Hint:* First show that the polynomial identity $x^m + 1 = (x^k + 1)(x^{k(t-1)} - x^{k(t-2)} + \cdots - x^k + 1)$ holds, where $m = kt$ and t is odd.]

***11.** Show that $\log_2 3$ is an irrational number. Recall that an irrational number is a real number x that cannot be written as the ratio of two integers.

12. Prove that for every positive integer n, there are n consecutive composite integers. [*Hint:* Consider the n consecutive integers starting with $(n + 1)! + 2$.]

***13.** Prove or disprove that there are three consecutive odd positive integers that are primes, that is, odd primes of the form p, $p + 2$, and $p + 4$.

14. Which positive integers less than 12 are relatively prime to 12?

15. Which positive integers less than 30 are relatively prime to 30?

16. Determine whether the integers in each of these sets are pairwise relatively prime.
a) 21, 34, 55 **b)** 14, 17, 85
c) 25, 41, 49, 64 **d)** 17, 18, 19, 23

17. Determine whether the integers in each of these sets are pairwise relatively prime.
a) 11, 15, 19 **b)** 14, 15, 21
c) 12, 17, 31, 37 **d)** 7, 8, 9, 11

18. We call a positive integer **perfect** if it equals the sum of its positive divisors other than itself.
a) Show that 6 and 28 are perfect.
b) Show that $2^{p-1}(2^p - 1)$ is a perfect number when $2^p - 1$ is prime.

19. Show that if $2^n - 1$ is prime, then n is prime. [*Hint:* Use the identity $2^{ab} - 1 = (2^a - 1) \cdot (2^{a(b-1)} + 2^{a(b-2)} + \cdots + 2^a + 1)$.]

20. Determine whether each of these integers is prime, verifying some of Mersenne's claims.
a) $2^7 - 1$ **b)** $2^9 - 1$
c) $2^{11} - 1$ **d)** $2^{13} - 1$

The value of the **Euler ϕ-function** at the positive integer n is defined to be the number of positive integers less than or equal to n that are relatively prime to n. [*Note:* ϕ is the Greek letter phi.]

21. Find these values of the Euler ϕ-function.
a) $\phi(4)$. **b)** $\phi(10)$. **c)** $\phi(13)$.

22. Show that n is prime if and only if $\phi(n) = n - 1$.

23. What is the value of $\phi(p^k)$ when p is prime and k is a positive integer?

24. What are the greatest common divisors of these pairs of integers?
a) $2^2 \cdot 3^3 \cdot 5^5$, $2^5 \cdot 3^3 \cdot 5^2$
b) $2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 \cdot 13$, $2^{11} \cdot 3^9 \cdot 11 \cdot 17^{14}$

c) $17, 17^{17}$ **d)** $2^2 \cdot 7, 5^3 \cdot 13$
e) $0, 5$ **f)** $2 \cdot 3 \cdot 5 \cdot 7, 2 \cdot 3 \cdot 5 \cdot 7$

25. What are the greatest common divisors of these pairs of integers?
a) $3^7 \cdot 5^3 \cdot 7^3, 2^{11} \cdot 3^5 \cdot 5^9$
b) $11 \cdot 13 \cdot 17, 2^9 \cdot 3^7 \cdot 5^5 \cdot 7^3$
c) $23^{31}, 23^{17}$
d) $41 \cdot 43 \cdot 53, 41 \cdot 43 \cdot 53$
e) $3^{13} \cdot 5^{17}, 2^{12} \cdot 7^{21}$
f) $1111, 0$

26. What is the least common multiple of each pair in Exercise 24?

27. What is the least common multiple of each pair in Exercise 25?

28. Find $\gcd(1000, 625)$ and $\text{lcm}(1000, 625)$ and verify that $\gcd(1000, 625) \cdot \text{lcm}(1000, 625) = 1000 \cdot 625$.

29. Find $\gcd(92928, 123552)$ and $\text{lcm}(92928, 123552)$, and verify that $\gcd(92928, 123552) \cdot \text{lcm}(92928, 123552) = 92928 \cdot 123552$. [*Hint:* First find the prime factorizations of 92928 and 123552.]

30. If the product of two integers is $2^7 3^8 5^2 7^{11}$ and their greatest common divisor is $2^3 3^4 5$, what is their least common multiple?

31. Show that if a and b are positive integers, then $ab = \gcd(a, b) \cdot \text{lcm}(a, b)$. [*Hint:* Use the prime factorizations of a and b and the formulae for $\gcd(a, b)$ and $\text{lcm}(a, b)$ in terms of these factorizations.]

32. Use the Euclidean algorithm to find
a) $\gcd(1, 5)$. **b)** $\gcd(100, 101)$.
c) $\gcd(123, 277)$. **d)** $\gcd(1529, 14039)$.
e) $\gcd(1529, 14038)$. **f)** $\gcd(11111, 111111)$.

33. Use the Euclidean algorithm to find
a) $\gcd(12, 18)$. **b)** $\gcd(111, 201)$.
c) $\gcd(1001, 1331)$. **d)** $\gcd(12345, 54321)$.
e) $\gcd(1000, 5040)$. **f)** $\gcd(9888, 6060)$.

34. How many divisions are required to find $\gcd(21, 34)$ using the Euclidean algorithm?

35. How many divisions are required to find $\gcd(34, 55)$ using the Euclidean algorithm?

***36.** Show that if a and b are both positive integers, then $(2^a - 1) \bmod (2^b - 1) = 2^{a \bmod b} - 1$.

***37.** Use Exercise 36 to show that if a and b are positive integers, then $\gcd(2^a - 1, 2^b - 1) = 2^{\gcd(a,b)} - 1$. [*Hint:* Show that the remainders obtained when the Euclidean algorithm is used to compute $\gcd(2^a - 1, 2^b - 1)$ are of the form $2^r - 1$, where r is a remainder arising when the Euclidean algorithm is used to find $\gcd(a, b)$.]

38. Use Exercise 37 to show that the integers $2^{35} - 1$, $2^{34} - 1$, $2^{33} - 1$, $2^{31} - 1$, $2^{29} - 1$, and $2^{23} - 1$ are pairwise relatively prime.

39. Using the method followed in Example 17, express the greatest common divisor of each of these pairs of integers as a linear combination of these integers.
a) 10, 11 **b)** 21, 44 **c)** 36, 48
d) 34, 55 **e)** 117, 213 **f)** 0, 223
g) 123, 2347 **h)** 3454, 4666 **i)** 9999, 11111

40. Using the method followed in Example 17, express the greatest common divisor of each of these pairs of integers as a linear combination of these integers.
a) 9, 11 **b)** 33, 44 **c)** 35, 78
d) 21, 55 **e)** 101, 203 **f)** 124, 323
g) 2002, 2339 **h)** 3457, 4669 **i)** 10001, 13422

The **extended Euclidean algorithm** can be used to express $\gcd(a, b)$ as a linear combination with integer coefficients of the integers a and b. We set $s_0 = 1, s_1 = 0, t_0 = 0$, and $t_1 = 1$ and let $s_j = s_{j-2} - q_{j-1}s_{j-1}$ and $t_j = t_{j-2} - q_{j-1}t_{j-1}$ for $j = 2, 3, \ldots, n$, where the q_j are the quotients in the divisions used when the Euclidean algorithm finds $\gcd(a, b)$, as shown in the text. It can be shown (see [Ro10]) that $\gcd(a, b) = s_n a + t_n b$. The main advantage of the extended Euclidean algorithm is that it uses one pass through the steps of the Euclidean algorithm to find Bézout coefficients of a and b, unlike the method in the text which uses two passes.

41. Use the extended Euclidean algorithm to express $\gcd(26, 91)$ as a linear combination of 26 and 91.

42. Use the extended Euclidean algorithm to express $\gcd(252, 356)$ as a linear combination of 252 and 356.

43. Use the extended Euclidean algorithm to express $\gcd(144, 89)$ as a linear combination of 144 and 89.

44. Use the extended Euclidean algorithm to express $\gcd(1001, 100001)$ as a linear combination of 1001 and 100001.

45. Describe the extended Euclidean algorithm using pseudocode.

46. Find the smallest positive integer with exactly n different positive factors when n is
a) 3. **b)** 4. **c)** 5.
d) 6. **e)** 10.

47. Can you find a formula or rule for the nth term of a sequence related to the prime numbers or prime factorizations so that the initial terms of the sequence have these values?
a) 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, ...
b) 1, 2, 3, 2, 5, 2, 7, 2, 3, 2, 11, 2, 13, 2, ...
c) 1, 2, 2, 3, 2, 4, 2, 4, 3, 4, 2, 6, 2, 4, ...
d) 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, ...
e) 1, 2, 3, 3, 5, 5, 7, 7, 7, 7, 11, 11, 13, 13, ...
f) 1, 2, 6, 30, 210, 2310, 30030, 510510, 9699690, 223092870, ...

48. Can you find a formula or rule for the nth term of a sequence related to the prime numbers or prime factorizations so that the initial terms of the sequence have these values?
a) 2, 2, 3, 5, 5, 7, 7, 11, 11, 11, 11, 13, 13, ...
b) 0, 1, 2, 2, 3, 3, 4, 4, 4, 4, 5, 5, 6, 6, ...
c) 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1, ...
d) 1, −1, −1, 0, −1, 1, −1, 0, 0, 1, −1, 0, −1, 1, 1, ...
e) 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 0, 0, ...
f) 4, 9, 25, 49, 121, 169, 289, 361, 529, 841, 961, 1369, ...

49. Prove that the product of any three consecutive integers is divisible by 6.

50. Show that if a, b, and m are integers such that $m \geq 2$ and $a \equiv b \pmod{m}$, then $\gcd(a, m) = \gcd(b, m)$.

***51.** Prove or disprove that $n^2 - 79n + 1601$ is prime whenever n is a positive integer.

52. Prove or disprove that $p_1 p_2 \cdots p_n + 1$ is prime for every positive integer n, where $p_1, p_2, \ldots, p_n$ are the n smallest prime numbers.

53. Show that there is a composite integer in every arithmetic progression $ak + b$, $k = 1, 2, \ldots$ where a and b are positive integers.

54. Adapt the proof in the text that there are infinitely many primes to prove that there are infinitely many primes of the form $3k + 2$, where k is a nonnegative integer. [*Hint:* Suppose that there are only finitely many such primes $q_1, q_2, \ldots, q_n$, and consider the number $3q_1 q_2 \cdots q_n - 1$.]

55. Adapt the proof in the text that there are infinitely many primes to prove that there are infinitely many primes of the form $4k + 3$, where k is a nonnegative integer. [*Hint:* Suppose that there are only finitely many such primes $q_1, q_2, \ldots, q_n$, and consider the number $4q_1 q_2 \cdots q_n - 1$.]

***56.** Prove that the set of positive rational numbers is countable by setting up a function that assigns to a rational number p/q with $\gcd(p, q) = 1$ the base 11 number formed by the decimal representation of p followed by the base 11 digit A, which corresponds to the decimal number 10, followed by the decimal representation of q.

***57.** Prove that the set of positive rational numbers is countable by showing that the function K is a one-to-one correspondence between the set of positive rational numbers and the set of positive integers if $K(m/n) = p_1^{2a_1} p_2^{2a_2} \cdot \cdots \cdot p_s^{2a_s} q_1^{2b_1 - 1} q_2^{2b_2 - 1} \cdot \cdots \cdot q_t^{2b_t - 1}$, where $\gcd(m, n) = 1$ and the prime-power factorizations of m and n are $m = p_1^{a_1} p_2^{a_2} \cdot \cdots \cdot p_s^{a_s}$ and $n = q_1^{b_1} q_2^{b_2} \cdots q_t^{b_t}$.

4.4 Solving Congruences

Introduction

Solving linear congruences, which have the form $ax \equiv b \pmod{m}$, is an essential task in the study of number theory and its applications, just as solving linear equations plays an important role in calculus and linear algebra. To solve linear congruences, we employ inverses modulo m. We explain how to work backwards through the steps of the Euclidean algorithm to find inverses modulo m. Once we have found an inverse of a modulo m, we solve the congruence $ax \equiv b \pmod{m}$ by multiplying both sides of the congruence by this inverse.

Simultaneous systems of linear congruence have been studied since ancient times. For example, the Chinese mathematician Sun-Tsu studied them in the first century. We will show how to solve systems of linear congruences modulo pairwise relatively prime moduli. The result we will prove is called the Chinese remainder theorem, and our proof will give a method to find all solutions of such systems of congruences. We will also show how to use the Chinese remainder theorem as a basis for performing arithmetic with large integers.

We will introduce a useful result of Fermat, known as Fermat's little theorem, which states that if p is prime and p does not divide a, then $a^{p-1} \equiv 1 \pmod{p}$. We will examine the converse of this statement, which will lead us to the concept of a pseudoprime. A pseudoprime m to the base a is a composite integer m that masquerades as a prime by satisfying the congruence $a^{m-1} \equiv 1 \pmod{m}$. We will also give an example of a Carmichael number, which is a composite integer that is a pseudoprime to all bases a relatively prime to it.

We also introduce the notion of discrete logarithms, which are analogous to ordinary logarithms. To define discrete logarithms we must first define primitive roots. A primitive root of a prime p is an integer r such that every integer not divisible by p is congruent to a power of r modulo p. If r is a primitive root of p and $r^e \equiv a \pmod{p}$, then e is the discrete logarithm of a modulo p to the base r. Finding discrete logarithms turns out to be an extremely difficult problem in general. The difficulty of this problem is the basis for the security of many cryptographic systems.

Linear Congruences

A congruence of the form

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

where m is a positive integer, a and b are integers, and x is a variable, is called a **linear congruence**. Such congruences arise throughout number theory and its applications.

How can we solve the linear congruence $ax \equiv b \pmod{m}$, that is, how can we find all integers x that satisfy this congruence? One method that we will describe uses an integer $\overline{a}$ such that $\overline{a}a \equiv 1 \pmod{m}$, if such an integer exists. Such an integer $\overline{a}$ is said to be an **inverse** of a modulo m. Theorem 1 guarantees that an inverse of a modulo m exists whenever a and m are relatively prime.

THEOREM 1 If a and m are relatively prime integers and $m > 1$, then an inverse of a modulo m exists. Furthermore, this inverse is unique modulo m. (That is, there is a unique positive integer $\overline{a}$ less than m that is an inverse of a modulo m and every other inverse of a modulo m is congruent to $\overline{a}$ modulo m.)

Proof: By Theorem 6 of Section 4.3, because $\gcd(a, m) = 1$, there are integers s and t such that

$$sa + tm = 1.$$

This implies that

$$sa + tm \equiv 1 \pmod{m}.$$

Because $tm \equiv 0 \pmod{m}$, it follows that

$$sa \equiv 1 \pmod{m}.$$

Consequently, s is an inverse of a modulo m. That this inverse is unique modulo m is left as Exercise 7. ◁

Using inspection to find an inverse of a modulo m is easy when m is small. To find this inverse, we look for a multiple of a that exceeds a multiple of m by 1. For example, to find an inverse of 3 modulo 7, we can find $j \cdot 3$ for $j = 1, 2, \ldots, 6$, stopping when we find a multiple of 3 that is one more than a multiple of 7. We can speed this approach up if we note that $2 \cdot 3 \equiv -1 \pmod{7}$. This means that $(-2) \cdot 3 \equiv 1 \pmod{7}$. Hence, $5 \cdot 3 \equiv 1 \pmod{7}$, so 5 is an inverse of 3 modulo 7.

We can design a more efficient algorithm than brute force to find an inverse of a modulo m when $\gcd(a, m) = 1$ using the steps of the Euclidean algorithm. By reversing these steps as in Example 17 of Section 4.3, we can find a linear combination $sa + tm = 1$ where s and t are integers. Reducing both sides of this equation modulo m tells us that s is an inverse of a modulo m. We illustrate this procedure in Example 1.

EXAMPLE 1 Find an inverse of 3 modulo 7 by first finding Bézout coefficients of 3 and 7. (Note that we have already shown that 5 is an inverse of 3 modulo 7 by inspection.)

Solution: Because $\gcd(3, 7) = 1$, Theorem 1 tells us that an inverse of 3 modulo 7 exists. The Euclidean algorithm ends quickly when used to find the greatest common divisor of 3 and 7:

$$7 = 2 \cdot 3 + 1.$$

From this equation we see that

$$-2 \cdot 3 + 1 \cdot 7 = 1.$$

This shows that -2 and 1 are Bézout coefficients of 3 and 7. We see that -2 is an inverse of 3 modulo 7. Note that every integer congruent to -2 modulo 7 is also an inverse of 3, such as 5, -9, 12, and so on. ◀

EXAMPLE 2 Find an inverse of 101 modulo 4620.

Solution: For completeness, we present all steps used to compute an inverse of 101 modulo 4620. (Only the last step goes beyond methods developed in Section 4.3 and illustrated in Example 17 in that section.) First, we use the Euclidean algorithm to show that $\gcd(101, 4620) = 1$. Then we will reverse the steps to find Bézout coefficients a and b such that $101a + 4620b = 1$. It will then follow that a is an inverse of 101 modulo 4620. The steps used by the Euclidean algorithm to find $\gcd(101, 4620)$ are

$$\begin{aligned}
4620 &= 45 \cdot 101 + 75 \\
101 &= 1 \cdot 75 + 26 \\
75 &= 2 \cdot 26 + 23 \\
26 &= 1 \cdot 23 + 3 \\
23 &= 7 \cdot 3 + 2 \\
3 &= 1 \cdot 2 + 1 \\
2 &= 2 \cdot 1.
\end{aligned}$$

Because the last nonzero remainder is 1, we know that $\gcd(101, 4620) = 1$. We can now find the Bézout coefficients for 101 and 4620 by working backwards through these steps, expressing $\gcd(101, 4620) = 1$ in terms of each successive pair of remainders. In each step we eliminate the remainder by expressing it as a linear combination of the divisor and the dividend. We obtain

$$\begin{aligned}
1 &= 3 - 1 \cdot 2 \\
&= 3 - 1 \cdot (23 - 7 \cdot 3) = -1 \cdot 23 + 8 \cdot 3 \\
&= -1 \cdot 23 + 8 \cdot (26 - 1 \cdot 23) = 8 \cdot 26 - 9 \cdot 23 \\
&= 8 \cdot 26 - 9 \cdot (75 - 2 \cdot 26) = -9 \cdot 75 + 26 \cdot 26 \\
&= -9 \cdot 75 + 26 \cdot (101 - 1 \cdot 75) = 26 \cdot 101 - 35 \cdot 75 \\
&= 26 \cdot 101 - 35 \cdot (4620 - 45 \cdot 101) = -35 \cdot 4620 + 1601 \cdot 101.
\end{aligned}$$

That $-35 \cdot 4620 + 1601 \cdot 101 = 1$ tells us that -35 and 1601 are Bézout coefficients of 4620 and 101, and 1601 is an inverse of 101 modulo 4620. ◀

Once we have an inverse $\overline{a}$ of a modulo m, we can solve the congruence $ax \equiv b \pmod{m}$ by multiplying both sides of the linear congruence by $\overline{a}$, as Example 3 illustrates.

EXAMPLE 3 What are the solutions of the linear congruence $3x \equiv 4 \pmod{7}$?

Solution: By Example 1 we know that -2 is an inverse of 3 modulo 7. Multiplying both sides of the congruence by -2 shows that

$$-2 \cdot 3x \equiv -2 \cdot 4 \pmod{7}.$$

Because $-6 \equiv 1 \pmod{7}$ and $-8 \equiv 6 \pmod{7}$, it follows that if x is a solution, then $x \equiv -8 \equiv 6 \pmod{7}$.

We need to determine whether every x with $x \equiv 6 \pmod{7}$ is a solution. Assume that $x \equiv 6 \pmod{7}$. Then, by Theorem 5 of Section 4.1, it follows that

$$3x \equiv 3 \cdot 6 = 18 \equiv 4 \pmod{7},$$

which shows that all such x satisfy the congruence. We conclude that the solutions to the congruence are the integers x such that $x \equiv 6 \pmod{7}$, namely, $6, 13, 20, \ldots$ and $-1, -8, -15, \ldots$. ◀

The Chinese Remainder Theorem

Systems of linear congruences arise in many contexts. For example, as we will see later, they are the basis for a method that can be used to perform arithmetic with large integers. Such systems can even be found as word puzzles in the writings of ancient Chinese and Hindu mathematicians, such as that given in Example 4.

Links

EXAMPLE 4 In the first century, the Chinese mathematician Sun-Tsu asked:

> There are certain things whose number is unknown. When divided by 3, the remainder is 2; when divided by 5, the remainder is 3; and when divided by 7, the remainder is 2. What will be the number of things?

This puzzle can be translated into the following question: What are the solutions of the systems of congruences

$$\begin{aligned} x &\equiv 2 \pmod{3}, \\ x &\equiv 3 \pmod{5}, \\ x &\equiv 2 \pmod{7}? \end{aligned}$$

We will solve this system, and with it Sun-Tsu's puzzle, later in this section. ◀

The *Chinese remainder theorem*, named after the Chinese heritage of problems involving systems of linear congruences, states that when the moduli of a system of linear congruences are pairwise relatively prime, there is a unique solution of the system modulo the product of the moduli.

THEOREM 2 **THE CHINESE REMAINDER THEOREM** Let $m_1, m_2, \ldots, m_n$ be pairwise relatively prime positive integers greater than one and $a_1, a_2, \ldots, a_n$ arbitrary integers. Then the system

$$\begin{aligned} x &\equiv a_1 \pmod{m_1}, \\ x &\equiv a_2 \pmod{m_2}, \\ &\vdots \\ x &\equiv a_n \pmod{m_n} \end{aligned}$$

has a unique solution modulo $m = m_1 m_2 \cdots m_n$. (That is, there is a solution x with $0 \leq x < m$, and all other solutions are congruent modulo m to this solution.)

Proof: To establish this theorem, we need to show that a solution exists and that it is unique modulo m. We will show that a solution exists by describing a way to construct this solution; showing that the solution is unique modulo m is Exercise 30.

To construct a simultaneous solution, first let

$$M_k = m/m_k$$

for $k = 1, 2, \ldots, n$. That is, M_k is the product of the moduli except for m_k. Because m_i and m_k have no common factors greater than 1 when $i \neq k$, it follows that $\gcd(m_k, M_k) = 1$. Consequently, by Theorem 1, we know that there is an integer y_k, an inverse of M_k modulo m_k, such that

$$M_k y_k \equiv 1 \pmod{m_k}.$$

To construct a simultaneous solution, form the sum

$$x = a_1 M_1 y_1 + a_2 M_2 y_2 + \cdots + a_n M_n y_n.$$

We will now show that x is a simultaneous solution. First, note that because $M_j \equiv 0 \pmod{m_k}$ whenever $j \neq k$, all terms except the kth term in this sum are congruent to 0 modulo m_k. Because $M_k y_k \equiv 1 \pmod{m_k}$ we see that

$$x \equiv a_k M_k y_k \equiv a_k \pmod{m_k},$$

for $k = 1, 2, \ldots, n$. We have shown that x is a simultaneous solution to the n congruences. ◁

Example 5 illustrates how to use the construction given in our proof of the Chinese remainder theorem to solve a system of congruences. We will solve the system given in Example 4, arising in Sun-Tsu's puzzle.

EXAMPLE 5 To solve the system of congruences in Example 4, first let $m = 3 \cdot 5 \cdot 7 = 105$, $M_1 = m/3 = 35$, $M_2 = m/5 = 21$, and $M_3 = m/7 = 15$. We see that 2 is an inverse of $M_1 = 35$ modulo 3, because $35 \cdot 2 \equiv 2 \cdot 2 \equiv 1 \pmod{3}$; 1 is an inverse of $M_2 = 21$ modulo 5, because $21 \equiv 1 \pmod{5}$; and 1 is an inverse of $M_3 = 15 \pmod{7}$, because $15 \equiv 1 \pmod{7}$. The solutions to this system are those x such that

$$\begin{aligned} x \equiv a_1 M_1 y_1 + a_2 M_2 y_2 + a_3 M_3 y_3 &= 2 \cdot 35 \cdot 2 + 3 \cdot 21 \cdot 1 + 2 \cdot 15 \cdot 1 \\ &= 233 \equiv 23 \pmod{105}. \end{aligned}$$

It follows that 23 is the smallest positive integer that is a simultaneous solution. We conclude that 23 is the smallest positive integer that leaves a remainder of 2 when divided by 3, a remainder of 3 when divided by 5, and a remainder of 2 when divided by 7. ◀

Although the construction in Theorem 2 provides a general method for solving systems of linear congruences with pairwise relatively prime moduli, it can be easier to solve a system using a different method. Example 6 illustrates the use of a method known as **back substitution**.

EXAMPLE 6 Use the method of back substitution to find all integers x such that $x \equiv 1 \pmod 5$, $x \equiv 2 \pmod 6$, and $x \equiv 3 \pmod 7$.

Solution: By Theorem 4 in Section 4.1, the first congruence can be rewritten as an equality, $x = 5t + 1$ where t is an integer. Substituting this expression for x into the second congruence tells us that

$$5t + 1 \equiv 2 \pmod 6,$$

which can be easily solved to show that $t \equiv 5 \pmod 6$ (as the reader should verify). Using Theorem 4 in Section 4.1 again, we see that $t = 6u + 5$ where u is an integer. Substituting this expression for t back into the equation $x = 5t + 1$ tells us that $x = 5(6u + 5) + 1 = 30u + 26$. We insert this into the third equation to obtain

$$30u + 26 \equiv 3 \pmod 7.$$

Solving this congruence tells us that $u \equiv 6 \pmod 7$ (as the reader should verify). Hence, Theorem 4 in Section 4.1 tells us that $u = 7v + 6$ where v is an integer. Substituting this expression for u into the equation $x = 30u + 26$ tells us that $x = 30(7v + 6) + 26 = 210u + 206$. Translating this back into a congruence, we find the solution to the simultaneous congruences,

$$x \equiv 206 \pmod{210}.$$ ◀

Computer Arithmetic with Large Integers

Suppose that $m_1, m_2, \ldots, m_n$ are pairwise relatively prime moduli and let m be their product. By the Chinese remainder theorem, we can show (see Exercise 28) that an integer a with $0 \leq a < m$ can be uniquely represented by the n-tuple consisting of its remainders upon division by m_i, $i = 1, 2, \ldots, n$. That is, we can uniquely represent a by

$$(a \bmod m_1, a \bmod m_2, \ldots, a \bmod m_n).$$

EXAMPLE 7 What are the pairs used to represent the nonnegative integers less than 12 when they are represented by the ordered pair where the first component is the remainder of the integer upon division by 3 and the second component is the remainder of the integer upon division by 4?

Solution: We have the following representations, obtained by finding the remainder of each integer when it is divided by 3 and by 4:

$0 = (0, 0)$ $4 = (1, 0)$ $8 = (2, 0)$
$1 = (1, 1)$ $5 = (2, 1)$ $9 = (0, 1)$
$2 = (2, 2)$ $6 = (0, 2)$ $10 = (1, 2)$
$3 = (0, 3)$ $7 = (1, 3)$ $11 = (2, 3)$.

◀