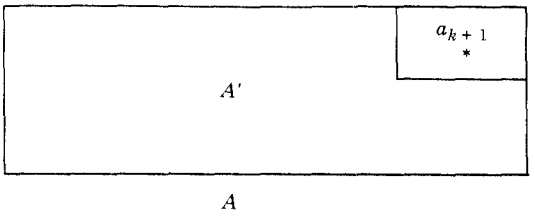


Figure 3.9

$\binom{k}{j-1}$ j-subsets of A that contain a_{k+1}. If S is a j-subset of A that does not contain a_{k+1}, then S is a j-subset of A'. By the inductive hypothesis there are $\binom{k}{j}$ of these subsets. Thus the total number of j-subsets of A is given by $\binom{k}{j-1}+\binom{k}{j}$. By Corollary 2.2 this sum equals $\binom{k+1}{j}$. □

Question 3.3. Use Theorem 3.1 to determine the number of k-subsets of an n-set, where (a) $k = 3$, $n = 6$; (b) $k = 5$, $n = 11$; and $k = 9$, $n = 17$.

Theorem 3.1 will enable us to prove a variety of results. For example, look back at Figure 3.7 and Exercise 2.5 and notice that the sum of the numbers in one row of Pascal's triangle seems to be a power of 2. It appears that for fixed n, the binomial coefficients of the form $\binom{n}{i}$ sum to 2^n. This result is an easy corollary of Theorem 3.1 once we look at the problem in the right way.

Corollary 3.2. For every natural number n

$$\binom{n}{0}+\binom{n}{1}+\cdots+\binom{n}{i}+\cdots+\binom{n}{n}=2^n.$$

Proof. As we've seen, the number of subsets of an n-set is 2^n. This number of subsets also equals the number of 0-subsets plus the number of 1-subsets plus $\cdots$ plus the number of i-subsets plus $\cdots$ plus the number of n-subsets. By Theorem 3.1 this sum is exactly the left-hand side of the equation and so the two expressions are equal. □

Example 3.2. Consider an n-digit number in binary. We can also think of this as an n-bit sequence of zeros and ones. How many n-bit sequences are there that contain exactly i ones, for i an integer between 0 and n? To answer this question, we need to recall a discussion about subsets and "bit vectors" of zeros and ones

presented in Chapter 1, Section 7 (see Question 7.5.) There it was shown that every n-bit sequence with i ones corresponds to a unique i-subset of an n-set, and conversely. Thus the number of n-bit sequences with i ones equals the number of i-subsets of an n-set, $\binom{n}{i}$ for $0 \leq i \leq n$.

Example 3.3. From Section 3.1 the number of shortest paths from M to P in the 6×5 grid equals the number of sequences of 11 Es and Ns with exactly 6 Es, and we know that this number equals $\binom{11}{6}$. Now we can see directly why there are $\binom{11}{6}$ sequences of 6 Es and 5 Ns. Suppose that we take any sequence of Es and Ns and replace every E by a one and every N by a zero. Then we have an 11-bit sequence with 6 ones; from Example 3.2 we know that there are $\binom{11}{6}$ of these.

Furthermore, every 11-bit sequence with 6 ones can be transformed back into a sequence of 6 Es and 5 Ns. In fact, Es and Ns or ones and zeros are two ways to represent exactly the same idea. Thus there are the same number of these two types of sequences.

Question 3.4. How many 11-letter sequences of Ns and Es are there that contain 3 Es?, 4 Es?, 7 Es? What is the total number of all possible 11-letter sequences of Es and Ns?

Problem. Given positive integers j and n with $j \leq n$, list all j-subsets of $\{1, 2, 3, \ldots, n\}$.

We know the number of j-subsets of an n-set without listing them all. However, we may wish to list all subsets of a certain size. We turn to the construction of an algorithm that upon input of natural numbers n and j with $j \leq n$ will list all j-subsets of an n-set. Up until now we've used the generic n-set $A = \{a_1, a_2, \ldots, a_n\}$. We simplify the notation and consider the n-set $I_n = \{1, 2, \ldots, n\}$ and its j-subsets. This change merely saves on writing: the j-subsets of A and I_n correspond exactly. We agree to always write the j-subset with its entries in increasing order.

Question 3.5. Describe how to transform a j-subset of A into a j-subset of I_n, and conversely. In particular, find the 4-subset of I_n that corresponds with $\{a_1, a_2, a_4, a_{n-1}\}$ and the 5-subset of A that corresponds with $\{1, 2, 3, 5, 8\}$.

For small, fixed values of j, the problem of creating an algorithm to list all j-subsets is not hard, and we begin with some specific cases. Exercise 14 asks you to design the straightforward algorithms to list all 0-subsets and all 1-subsets of $I_n = \{1, 2, \ldots, n\}$.

Example 3.4. Here is an algorithm that lists all 2-subsets of $I_n = \{1, 2, \ldots, n\}$. (This also was Exercise 1.7.6.)

Algorithm PAIR

STEP 1. Input n, a positive integer
STEP 2. For $i = 1$ to $(n - 1)$ do
Begin
STEP 3. For $j = (i + 1)$ to n do
STEP 4. List $\{i,j\}$
End
STEP 5. Stop.

COMMENTS. Section 2.4 contained examples of this form of a loop, often called a "**do loop.**" Recall that the instructions, "For $i = 1$ to $(n - 1)$ do, Begin, . . . , End," mean that first i should be set equal to 1 and the instructions between the Begin and End statements should be executed. Next, i is set equal to 2 and the Begin . . . End statements are executed. We repeat these steps, increasing i by one each time until i is set equal to $(n - 1)$, the Begin . . . End statements are executed for the last time, and then we move on to the next statement, here step 5. This is also our first example of a loop within a loop; such pairs of loops are called **nested**. For each assignment of a value to i, step 3 is repeated several times, once for each of the values $j = (i + 1), (i + 2), \ldots$, and n. Then i is assigned its next value and j runs through its sequence of values.

The output of PAIR has the following properties. The subset $\{i,j\}$ is always listed with its smaller element first. The first subsets listed are those that contain 1; next the subsets that contain 2 (but not 1) are listed; then those with 3; and so on. Here is how the output looks:

$$\{1,2\},\ \{1,3\},\ \{1,4\},\ \ldots,\ \{1,n\},$$
$$\{2,3\},\ \{2,4\},\ \ldots,\ \{2,n\},$$
$$\{3,4\},\ \ldots,\ \{3,n\},$$
$$\ldots$$
$$\{n-2,n-1\},\ \{n-2,n\},$$
$$\{n-1,n\}.$$

This ordering is known as **lexicographic** or **dictionary order**.

The idea behind PAIR could be generalized to generate all 3-subsets of an n-set by having 3 nested loops. However, if we want to list all j-subsets, where j is part of the input to the algorithm, then this idea won't work, since we can't have the correct number of loops programmed in advance. Instead we generalize the idea of lexicographic order for arbitrary $j > 2$ and use this order to list all j-subsets.

We approach lexicographic ordering in two ways. First we describe to you, a human, which of any two j-subsets of an n-set should be listed before the other. Next we need to explain this order to a computer, that is, in language sufficiently precise so that it can be implemented as an algorithm. We need to know exactly how, once the algorithm has created and listed a j-subset S, to move on and produce the next j-subset T. At the same time we must explain why lexicographic order as described for humans is the same as the order described for computers.

We begin our description for humans with an example.

Example 3.5. Here are the 3-subsets of $I_5 = \{1,2,3,4,5\}$ listed in lexicographic order: $\{1,2,3\}, \{1,2,4\}, \{1,2,5\}, \{1,3,4\}, \{1,3,5\}, \{1,4,5\}, \{2,3,4\}, \{2,3,5\}, \{2,4,5\}, \{3,4,5\}$.

One way to think about lexicographic order is to disregard all the set notation that accompanies each subset and just list the integers in the order in which they appear. Then we see

$$123 < 124 < 125 < 134 < 135 < 145 < 234 < 235 < 245 < 345,$$

and the ordering is the same as the "less than" order on the natural numbers.

Question 3.6. Place the following sets of I_8 in lexicographic order. $\{1,3,4,7\}$, $\{3,6,7,8\}$, $\{2,3,4,7\}$, $\{4,6,7,8\}$, $\{2,3,5,6\}$, and $\{1,3,5,6\}$.

This description would suffice for humans and computers except for the fact that if $n > 9$, we might see sets like $\{2,3,45,678\}$ and $\{2,34,56,78\}$. This could be salvaged if we were willing to do arithmetic in base n, but since n is part of the input, there are still problems with this approach.

Here is a way for humans and computers to decide which of two j-subsets, S and T, to list first. Suppose that $S = \{s_1, \ldots, s_j\}$ and $T = \{t_1, \ldots, t_j\}$. If $s_1 < t_1$, then S is listed before T. If $s_1 > t_1$, then T is listed before S. The only other alternative is that $s_1 = t_1$ in which case we compare the second element of each set. In general, if $s_1 = t_1, s_2 = t_2, \ldots, s_{i-1} = t_{i-1}$, and $s_i < t_i$, then we list S before T, and if $s_i > t_i$ we list T before S. In English, S appears before T if in the first place the two sets differ, the element of S is smaller than the element of T.

Example 3.6. $\{2,3,4,7,8,9\}$ is listed after $\{2,3,4,6,8,9\}$, since the sets agree until the fourth entry at which point $7 > 6$.

When listing all subsets of a set, if S is listed immediately before T, we should expect that S and T agree as much as possible. Furthermore, we should expect that at the location where S and T first disagree, the entry in T is just one larger than the corresponding entry in S.

Example 3.5 (revisited). A second way to view lexicographic ordering of the 3-subsets of $\{1,2,3,4,5\}$ is to start with the "first" set $\{1,2,3\}$. Then we let the last entry increase by 1 and we get $\{1,2,4\}$. Repeating this, we get $\{1,2,5\}$. So our listing begins

$$\{1,2,3\}, \quad \{1,2,4\}, \quad \{1,2,5\}.$$

Now the last entry is as big as it can be. So we agree to increase the second to last entry by 1 and make every other entry as small as it can be. Thus after $\{1,2,5\}$ we get $\{1,3,4\}$. Now we return our attention to the last entry, which can once again be incremented. Thus the list continues

$$\{1,3,4\}, \quad \{1,3,5\}, \quad \{1,4,5\}.$$

Now our second to last entry is as big as it can be, so we have to increment our third to last (here the same as the first) entry. We finish with

$$\{2,3,4\}, \quad \{2,3,5\}, \quad \{2,4,5\}, \quad \{3,4,5\}.$$

Question 3.7. List all 3-subsets of $I_6 = \{1,2,3,4,5,6\}$ in lexicographic order.

Thus, in general, the idea of lexicographic order is to begin with the j-subset $\{1,2,\ldots,j-1,j\}$ and then to let the last entry increase through all possible values until we reach the subset $\{1,2,\ldots,j-1,n\}$. Next we move back to the second to last entry, increase it by one, and let the last entry take on all larger values. In general, after a subset $S = \{s_1,\ldots,s_h,\ldots,s_j\}$ is listed, we search from right to left in S looking for the first entry that can be increased by one. Suppose that s_h can be increased to $s_h + 1$, but no s_k with $k > h$ can be increased. Then we produce the new subset $T = \{s_1,\ldots,s_{h-1}, s_h + 1, s_h + 2, s_h + 3,\ldots\}$.

Example 3.7. Here is how these ideas apply when listing all 4-subsets of $\{1,2,\ldots,8\}$ in lexicographic order.

$$\begin{array}{llll}
\{1,2,3,4\}, & \{1,2,3,5\}, & \ldots, & \{1,2,3,8\}, \\
\{1,2,4,5\}, & \{1,2,4,6\}, & \ldots, & \{1,2,4,8\}, \\
\{1,2,5,6\}, & \{1,2,5,7\}, & \{1,2,5,8\}, & \\
\{1,2,6,7\}, & \{1,2,6,8\}, & \{1,2,7,8\}, & \\
\{1,3,4,5\}, & \{1,3,4,6\}, & \ldots, & \{1,3,4,8\}, \\
\{1,3,5,6\}, & \ldots, \{1,6,7,8\}, & & \\
\{2,3,4,5\}, & \{2,3,4,6\}, & \ldots, & \{5,6,7,8\}.
\end{array}$$

We know that we are finished, since no entry can be increased.

Question 3.8. List all 3-subsets of $I_7 = \{1, 2, \ldots, 7\}$ in lexicographic order.

Notice that in Example 3.1 if the set $I_6 = \{1, 2, 3, 4, 5, 6\}$ is substituted for A and the corresponding 3-subsets of A are replaced by those of I_6, then the subsets are listed in lexicographic order.

This work gives us the ideas for an algorithm to list all j-subsets of an n-set for a general value of j and n. The only imprecision left is how, after a j-subset S is listed, to find the entry s_h that can be increased to $s_h + 1$. Note that the jth entry in a j-subset can be no larger than n, the $(j-1)$st entry can be no larger than $n-1$, and in general the ith entry can be no larger than $n + i - j$. Thus we read from right to left until we find s_h such that $s_h < n + h - j$. We replace s_h by $s_h + 1$ and fill in the remaining subset entries with $s_h + 2$, $s_h + 3, \ldots$. The algorithm stores each subset in an array $\{b_1, b_2, \ldots, b_j\}$.

Algorithm JSET

STEP 1. Input j and n with $1 \leq j \leq n$
STEP 2. Set $\{b_1, b_2, \ldots, b_j\} := \{1, 2, \ldots, j\}$ and list this set
STEP 3. Set $h := j + 1$ and FOUND := FALSE
{Next we determine which entry to increase by 1. When we find the entry, we set FOUND to be TRUE.}
STEP 4. While $h > 1$ and FOUND = FALSE do
Begin
STEP 5. $h := h - 1$
STEP 6. If $b_h < n + h - j$, set FOUND := TRUE
End
{Right now the value assigned to h is the rightmost entry that can be increased. If FOUND = FALSE, then we are finished listing all subsets.}
STEP 7. If FOUND = FALSE, then stop.
Else
Begin
STEP 8. Set $b_h := b_h + 1$
STEP 9. For $k := h + 1$ to j do
STEP 10. Set $b_k := b_{k-1} + 1$
STEP 11. List $\{b_1, b_2, \ldots, b_j\}$
STEP 12. Go to step 3
End

COMMENTS. We introduce the so-called Boolean variable FOUND. (See also Section 10 of Chapter 1.) A **Boolean variable** can be assigned only two values, either TRUE or FALSE. We initialize FOUND to be FALSE and change its value if the correct entry to increment has been located. Before repeating the steps inside

the While do loop at step 4, two conditions must be satisfied. It must be the case that both $h > 1$ and FOUND = FALSE. If either condition is contradicated, then we proceed to step 7. In general, Boolean variables are useful in the design of loops when the number of iterations through the loop is not known in advance.

With $j = 3$ and $n = 5$, JSET lists the subsets as shown in Example 3.5 and with $j = 4$ and $n = 8$ as in Example 3.7.

Question 3.9. Run JSET with $j = 4$ and $n = 6$.

We conclude with an informal analysis of the complexity of JSET. The input is j and n with $j \leq n$; so we use n as a measure of the input size. We let $f(n)$ equal the maximum number of time-consuming operations performed in JSET and carry out a worst-case analysis. We hope to find a simple function $g(n)$ [like $\log(n)$, n^2, or 2^n] such that $f(n) = O(g(n))$. Instead of counting all the additions, assignments, and comparisons in JSET, we choose to define a basic, time-consuming operation to be the creation and listing of a j-subset. (For a more precise analysis see Exercise 23.) Since $\binom{n}{j}$ j-subsets are produced, $f(n) = \binom{n}{j}$. Is JSET good? That depends on the value of j. For example, when $j = 2$, the complexity of JSET is

$$f(n) = \binom{n}{2} = O(n^2)$$

and in this case JSET is a good algorithm.

Question 3.10. Show that JSET is a good algorithm when $j = 3$.

Question 3.11. Show that if j is a fixed constant less than $n/2$, then

$$\binom{n}{j} = O(n^j).$$

However, there are cases in which JSET is an exponential algorithm. We have seen in Section 3.2 that $\binom{n}{j}$ is largest when $j = \lfloor n/2 \rfloor$. We shall show in the next theorem that $\binom{n}{\lfloor n/2 \rfloor}$ is exponential in n [thus not $O(p(n))$ for any polynomial p].

Does that mean that we have failed to be sufficiently clever in our algorithm design? No, because when $j = \lfloor n/2 \rfloor$, there is an exponential number of j-subsets, and any j-subset algorithm will be exponential, since each of these must be listed. Thus although this algorithm is exponential and not good, it is nonetheless more or less as efficient as possible and so we'll use it when needed.

Theorem 3.3 For n even $\binom{n}{n/2} \geq (\sqrt{2})^n$.

Proof

$$\begin{aligned}\binom{n}{n/2} &= \frac{n!}{(n/2)!\,(n-n/2)!} = \frac{n!}{(n/2)!\,(n/2)!} \\ &= \frac{n(n-1)(n-2)\cdots(n/2+1)}{(n/2)(n/2-1)(n/2-2)\cdots 2\cdot 1} \\ &= \frac{n}{n/2}\cdot\frac{n-1}{n/2-1}\cdot\cdots\cdot\frac{n-j}{n/2-j}\cdot\cdots\cdot\frac{n/2+1}{1}.\end{aligned}$$

Next we consider an arbitrary factor in the above expression.

$$\frac{n-j}{n/2-j} = \frac{n-j}{\frac{(n-2j)}{2}} = 2\cdot\frac{n-j}{n-2j} \geq 2.$$

Thus each of these factors is at least 2. Since there are $n/2$ factors, the product is at least $2^{n/2} = (\sqrt{2})^n$. □

EXERCISES FOR SECTION 3

1. For $i = 0, 1, \ldots, 6$ calculate the number of i-subsets of a 6-set. Check that these numbers sum to 2^6.
2. For $i = 0, 2, 4, 6, 8$ calculate the number of i-subsets of an 8-set. Check that these numbers sum to 2^7.
3. The number of j-subsets of an n-set equals $\binom{n}{j}$, and the number of $(n-j)$-subsets equals $\binom{n}{n-j}$. Give a set theory argument to show that these two numbers are equal.
4. From a class of 14 students, a committee of 5 students is formed. In how many different ways can this committee be chosen? How many of these committees contain one particular student, Sue, and how many committees do not contain her?
5. Suppose that a class is composed of eight freshmen and six sophomores. How many committees are there that consist of three freshmen and two sophomores?

6. Suppose that two committees with no members in common are to be selected from a class of 14 students. First a committee of 5 students is selected and then a committee of 3 students. In how many ways can these two committees be formed? Is the answer to the last question the same as the number of ways to form first a committee of 3 students and then second a committee of 5 students?

7. The local Pizza Factory produces pizzas topped with tomato sauce and cheese. Optional items are mushrooms, green peppers, olives, onions, anchovies, pepperoni, and hamburger. The prices are \$3.00 for plain pizza, \$3.50 for one extra item, \$4.00 for two items, and \$4.50 for three. How many different kinds of pizza cost \$3.50? \$4.00? \$4.50?

8. The ice cream parlor in town offers 10 different kinds of ice cream daily plus toppings of fudge sauce, butterscotch sauce, raspberry sauce, walnuts, M&Ms, Heath Bar Crunch, and whipped cream. How many different ways are there to make a sundae if (by definition) a sundae consists of one kind of ice cream plus at least one topping? How many kinds of sundae are there that have exactly 3 toppings? That have no more than 3 toppings?

9. Prove by induction on n that

$$\binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n} = 2^n.$$

10. If n is even, show that

$$\binom{n}{0} + \binom{n}{2} + \binom{n}{4} + \cdots + \binom{n}{n} = 2^{n-1}.$$

Then find and justify a similar formula when n is odd.

11. List all seven-bit vectors with exactly two ones. Then list all sequences of Es and Ns that contain exactly five Ns and two Es.

12. For $j = 2, 3, \ldots, 7$ calculate the binomial coefficients $\binom{j}{2}$ and find their sum. Then find a formula in terms of n that compactly expresses the sum

$$\binom{2}{2} + \binom{3}{2} + \cdots + \binom{n}{2}.$$

13. Design an algorithm to create and list all bit vectors of length n.

14. Describe an algorithm that will list all 0-subsets of $I_n = \{1, 2, \ldots, n\}$. Then describe an algorithm that will list all 1-subsets of I_n.

15. (*a*) Modify the algorithm PAIR to list all bit vectors of length n that contain exactly 2 ones.

(*b*) Modify PAIR to list all sequences of Es and Ns that contain exactly 2 Es and $n - 2$ Ns.

16. How many 2-subsets of an n-set contain the element 1? How many 2-subsets contain the element 2 but not 1? How many contain 3 but neither 1 nor 2? In general, how many 2-subsets contain the element i but no smaller number? Then add up all these answers for $i = 1, 2, \ldots, (n-1)$ and explain why that sum adds up to $\binom{n}{2}$, the total number of 2-subsets of an n-set.

17. List all 2-subsets of $\{1, 2, \ldots, 6\}$ in lexicographic order. Then list all 5-subsets of the set $\{1, 2, 3, 4, 5, 6, 7\}$ in lexicographic order.

18. We defined lexicographic order of subsets of numbers by saying that we list a subset S before a subset T "if in the first place the two sets differ, the element of S is smaller than the element of T." If we consider now subsets of a set $A = \{a, b, c, d, e, f\}$ and use alphabetical ordering of letters, how should we define lexicographic order of subsets of A? List all 2-subsets of A in lexicographic order. Then check that in your list, when a is replaced by 1, b by 2, ..., and f by 6, that the 2-subsets are still listed in lexicographic order (for numbers).

19. Run the algorithm JSET with
(*a*) $j = 1$ and $n = 6$.
(*b*) $j = 3$ and $n = 6$.
(*c*) $j = 3$ and $n = 7$.

20. Is JSET a good algorithm when $j = n - 2$? When $j = n - 1$?

21. If n is odd, show that

$$\binom{n}{\frac{(n-1)}{2}} \geq r^n \qquad \text{for some } r > 1.$$

22. Show that

$$\binom{n}{j} = O(2^n) \qquad \text{for all } j.$$

23. Give a more detailed, worst-case complexity analysis of JSET by counting the number of additions, subtractions, comparisons, and assignments. Note that the While do loop of step 4 is executed at most j times and the For do loop of step 9 at most $(j - 1)$ times. Show that the total number of these operations is $O\left(j\binom{n}{j}\right)$. Then explain why this analysis shows JSET to be a polynomial algorithm for j a constant number, but for j arbitrary JSET is exponential.

3:4 PERMUTATIONS

When we were generating all the j-subsets of the n-set I_n in Section 3, it was crucial to be able to arrange the elements within each subset in increasing order. The reason we could do this is that in sets, the order of presentation of the elements does not matter and so we could choose an algorithmically useful presentation. In contrast we have seen some cases earlier where the given order is important and we cannot rearrange elements. Examples include applications of the Multiplication Principle and Cartesian products. We turn now to another area where order is crucial.

Given a set $A = \{a_1, a_2, \ldots, a_n\}$ containing n distinct elements, an ordered list of these n elements is called a **permutation** of A. Often, but not always, we shall use the set $I_n = \{1, 2, \ldots, n\}$ as the generic set with n elements. We distinguish sets, which will continue to be denoted by $\{\cdots\}$, from permutations, which will be surrounded by $\langle\cdots\rangle$.

Example 4.1. If $I_6 = \{1,2,3,4,5,6\}$, then $\langle 1\ 2\ 3\ 4\ 5\ 6\rangle$, $\langle 2\ 1\ 4\ 3\ 6\ 5\rangle$, and $\langle 1\ 3\ 5\ 2\ 4\ 6\rangle$ are three permutations of I_6; however, $\langle 1\ 2\ 4\ 5\rangle$ is not a permutation of I_6, neither is $\langle 1\ 2\ 3\ 4\ 5\ 5\rangle$ nor $\langle 1\ 2\ 3\ 4\ 5\ 7\rangle$.

Within the permutation $\langle 2\ 1\ 3\ 5\ 4\ 6\rangle$ we say that 2 is in the first position, 1 in the second position, 3 in the third position, and so on.

Example 4.2. Here are all permutations on $\{1,2,3\}$:

$$\langle 1\ 2\ 3\rangle,\quad \langle 1\ 3\ 2\rangle,\quad \langle 3\ 1\ 2\rangle,\quad \langle 2\ 1\ 3\rangle,\quad \langle 2\ 3\ 1\rangle,\quad \langle 3\ 2\ 1\rangle.$$

Notice there are $6 = 3!$ such permutations.

Question 4.1. List all 4! permutations on the four numbers $\{1,2,3,4\}$.

The Multiplication Principle shows us why the factorial function counts permutations of an n-set. Since a permutation is an ordered list, we may count the ways to fill in the n blanks:

$$\langle \_\ \_\ \_\ \cdots\ \_\rangle$$

First we see that we have n choices for the element to place in the first position of the permutation. After that we have only $(n-1)$ choices for the entry in the second position, then $(n-2)$ for the third position, and so on. At the next to last position we have two choices remaining and then one choice for the last position. Thus the number of permutations on n elements is

$$n(n-1)(n-2)\cdots 2\cdot 1 = n!.$$

This argument proves the next theorem. One proof is sufficient for any theorem, but we choose to present a second proof using induction because it will cause us to think about generating permutations algorithmically.

Theorem 4.1. There are $n!$ permutations of a set containing n distinct elements.

Proof. First we check the base cases for $n = 1$ and 2: $\langle 1 \rangle$ is the only permutation on one number and $\langle 1\ 2 \rangle$ and $\langle 2\ 1 \rangle$ are the two permutations on two numbers. In Example 4.2 and Question 4.1 we checked the cases for $n = 3$ and $n = 4$.

Our inductive hypothesis is that there are $k!$ permutations of any set with k elements. Using this hypothesis, we need to show that any set with $k + 1$ elements has $(k + 1)!$ permutations. Let $I_{k+1} = \{1, 2, \ldots, k, k + 1\}$. As usual we remove an element, setting $I_k = I_{k+1} - \{k + 1\} = \{1, 2, \ldots, k\}$. Pick any permutation of I_k: for instance,

$$\langle 2\ 1\ 3\ 4 \cdots k \rangle.$$

Each such permutation of I_k generates $k + 1$ different permutations of I_{k+1} because the element $(k + 1)$ can be inserted in $k + 1$ different positions. From the permutation above we create

$$\begin{array}{llllllll}
\langle & 2 & 1 & 3 & \cdots & k & k+1 & \rangle, \\
\langle & 2 & 1 & 3 & \cdots & k+1 & k & \rangle, \\
 & & & & \cdots & & & \\
\langle & 2 & 1 & 3 & k+1 & \cdots & k & \rangle, \\
\langle & 2 & 1 & k+1 & 3 & \cdots & k & \rangle, \\
\langle & 2 & k+1 & 1 & 3 & \cdots & k & \rangle,
\end{array}$$

and

$$\langle k+1 \quad 2 \quad 1 \quad 3 \quad \cdots \quad k \rangle.$$

Thus the number of permutations of I_{k+1} equals $k + 1$ times the number of permutations of I_k, or $(k + 1)(k!) = (k + 1)!$. □

Remember this proof because it suggests how to write an algorithm to generate all permutations.

Example 4.3. Suppose that we start with the permutation $\langle 2\ 1\ 3\ 4 \rangle$. We can insert the element 5 in any one of five different positions to obtain $\langle 2\ 1\ 3\ 4\ 5 \rangle$, $\langle 2\ 1\ 3\ 5\ 4 \rangle$, $\langle 2\ 1\ 5\ 3\ 4 \rangle$, $\langle 2\ 5\ 1\ 3\ 4 \rangle$, and $\langle 5\ 2\ 1\ 3\ 4 \rangle$.

Permutations give us another way to see that an n-set has $\binom{n}{j}$ j-subsets for $0 \leq j \leq n$. A 2-subset of the set $I_n = \{1, 2, \ldots, n\}$ can be formed by filling in two blanks: $\{\_, \_\}$. By the Multiplication Principle there are n choices for the first blank and $(n-1)$ choices for the second; however, once the set is filled in, every permutation of the elements in the set will give the same set. On two elements there are $2! = 2$ permutations and so in the $n(n-1)$ choices, each set is listed twice. Thus there are $n(n-1)/2 = \binom{n}{2}$ 2-subsets.

Question 4.2. Use the Multiplication Principle and permutations to show that there are $n(n-1)(n-2)/3!$ different 3-subsets of an n-set.

Next consider the case of counting the number of j-subsets of an n-set for an arbitrary value of j between 0 and n. To create a j-subset, begin by filling in j empty blanks: $\{\_, \_, \ldots, \_\}$. There are n choices for the first blank, then $(n-1)$ choices for the second blank, and eventually for the jth blank, there are $n - (j-1) = n - j + 1$ choices remaining. Thus there are

$$n(n-1)\cdots(n-j+1) = \frac{n!}{(n-j)!}$$

choices for filling in j empty blanks. But each subset with j elements can arise in $j!$ different orders, one for each of the permutations on j elements. Thus the total number of subsets of size j is given by

$$\frac{n!}{(n-j)!j!} = \binom{n}{j}.$$

Now we turn to the algorithmic question of generating all permutations. We displayed all permutations on three elements in Example 4.2, and you were asked to do the same for four elements in Question 4.1. For $n > 4$, $n!$ is larger than 100, and listing all permutations is clearly a task suitable for computers rather than humans.

Problem. Given an integer $n > 0$, list all permutations of $\{1, \ldots, n\}$.

Look back at the proof of Theorem 4.1. It tells us that from each permutation on $(n-1)$ elements, we get n permutations on n elements by moving the element n through all possible positions in the $(n-1)$ permutation. To use this idea, we need to begin at the base case of our inductive proof. We first write down all permutations on one element: $\langle 1 \rangle$. Then we use the idea of the inductive step to generate all permutations on two elements: $\langle 1\ 2 \rangle$ and $\langle 2\ 1 \rangle$. How about three elements? First we expand $\langle 1\ 2 \rangle$ into the three permutations $\langle 1\ 2\ 3 \rangle$, $\langle 1\ 3\ 2 \rangle$ and $\langle 3\ 1\ 2 \rangle$.

Then we expand the permutation $\langle 2\ 1\rangle$ into $\langle 2\ 1\ 3\rangle$, $\langle 2\ 3\ 1\rangle$, and $\langle 3\ 2\ 1\rangle$. Here is an algorithm that works exactly in this way.

Algorithm PERM

STEP 1. Set $j := 1$ and write down the permutation $\langle 1\rangle$
STEP 2. Set $j := j + 1$
STEP 3. For each permutation $\langle a_1\, a_2 \ldots a_{j-1}\rangle$ on $j - 1$ elements do
Begin
STEP 4. Create and list $P := \langle a_1 a_2 \ldots a_{j-1} j\rangle$
STEP 5. For $i = j - 1$ down to 1 do
STEP 6. Set $P := P$ with the values assigned to positions i and $(i + 1)$ switched, and list P
End {step 3}
STEP 7. If $j < n$, then go to step 2; otherwise, stop.

COMMENTS. There are some new features in this algorithm and some old. The first new feature is the form of step 5, "**For** $i = j - 1$ **down to** 1 **do**." This indicates a do-loop; here the variable i will decrease from $j - 1$. Thus initially i equals $j - 1$ and step 6 is executed; then i equals $j - 2$ for step 6, . . . , until finally i equals 1 for the last execution of step 6, at least for this round.

We have previously seen an example (in PAIR) of a "nested loop," a loop within a loop. Step 3 is an instruction that causes steps 4 and 5 to be executed for each permutation on $j - 1$ elements. Also, step 5 instructs us to carry out step 6 $j - 1$ times for each time step 5 is encountered. Actually, the algorithm also has a third loop that sends us back to step 2 repeatedly. We do not specify step 6 precisely, but notice that it requires the switching of the values of two variables. exactly the task that we learned to do in Section 2.1.

Example 4.4. In Table 3.1 we trace through algorithm PERM in the case of $n = 3$

Table 3.1

Values of j	*Permutations*
1	$\langle 1\rangle$
2	$\langle 1\ 2\rangle$
	$\langle 2\ 1\rangle$
3	$\langle 1\ 2\ 3\rangle$
	$\langle 1\ 3\ 2\rangle$
	$\langle 3\ 1\ 2\rangle$
	$\langle 2\ 1\ 3\rangle$
	$\langle 2\ 3\ 1\rangle$
	$\langle 3\ 2\ 1\rangle$

Question 4.3. Use Algorithm PERM to generate all permutations on $I_4 = \{1, 2, 3, 4\}$.

Our next task is to analyze the complexity of PERM. We do even fewer kinds of time-consuming operations than in JSET. The algorithm mainly assigns, reassigns, and lists permutations, $n!$ of these, in fact.

To analyze PERM, we need to study the factorial function and relate it to our hierarchy of functions: we know that for sufficiently large n,

$$1 \leq \log(n) \leq \sqrt{n} \leq n \leq n^2 \leq n^3 \leq \cdots \leq 2^n \leq \cdots \leq 10^n.$$

We shall find that not only is $n! \neq O(p(n))$ for any polynomial n, but even more, in Theorem 4.2 we prove that for any positive integer $r > 0$ and any constant C, $n! > C \cdot r^n$ for n sufficiently large. Thus the correct position for $n!$ in the hierarchy is at the end (so far):

$$\cdots \leq 10^n \leq \cdots \leq n!.$$

We count the formation and listing of each permutation as the fundamental time-consuming operation. PERM performs $n!$ of these steps and is thus not a polynomial algorithm. However, PERM accomplishes even more than it was asked to do. Besides generating all permutations on $\{1, 2, \ldots, n\}$, it also generates all permutations on the smaller sets $\{1, 2, \ldots, i\}$ for each $i = 1, 2, \ldots, n-1$. Thus PERM really performs $1! + 2! + \cdots + n!$ steps and its complexity is consequently even larger than $n!$.

Also notice that in PERM we must store all $(i-1)!$ permutations on $i-1$ numbers before creating the permutations on i numbers. Hence this algorithm requires more than an exponential amount of storage as well as more than an exponential amount of time, and so it is bad with respect to both time and space, the worst of both worlds.

Generating permutations is always time-consuming; however, there are ways to generate permutations on n elements without first generating all permutations on smaller sets. Also there are ways to create a "next" permutation from a given permutation (as in the algorithm JSET) so that in an application the permutations can be produced one after the other without storing previous permutations. Some of the ideas behind these algorithms are presented in the exercises.

But why is $n!$ such a fast growing function? Why does it belong at the end of the function hierarchy? Suppose that we add the function $n!$ to Table 2.9 in Chapter 2. That is, suppose that we perform $n!$ single digit multiplications on an IBM PC. The time involved for differing values of n is listed in Table 3.2.

The numerical evidence is clear that the function $n!$ grows even faster than 10^n. Suppose as with multiplication on the PC, we could figure out and write

Table 3.2

	$n = 8$	$n = 11$	$n = 16$	$n = 24$	$n = 32$
$n!$	2.27 min	37.4 hr	2236 yr	663,178,306,400 cent	⋯

down 17,800 permutations per minute (and this would be working pretty fast!) Then Table 3.2 tells us that using PERM in, say, a 24-hour run we could only hope to list all permutations for $n \leq 10$.

Here is theoretical evidence that also points to the rapid growth of $n!$.

Theorem 4.2. $n! \neq O(r^n)$ for any positive integer r.

Proof. The theorem states that it is not the case that given a positive integer r, there is a constant C such that $n! \leq C \cdot r^n$ for all sufficiently large integers n. Thus given an integer r we must show that for every constant C, $n! > C \cdot r^n$ for n sufficiently large. So pick a constant C and suppose that $n > C \cdot r^r$. (This number is cooked up because we found that it made the proof work easily!) We know that $r = r$ and $r + 1 > r$, and $r + 2 > r$, and $\ldots, n - 1 > r$. Thus we have

$$\begin{aligned} n! &= n(n-1)\cdots(r+1)r\cdots 2\cdot 1 && \text{by definition} \\ &\geq n(n-1)\cdots(r+1)r && \text{by ignoring the last factors} \\ &> n\cdot r\cdot\cdots\cdot r\cdot r && \text{(a product of } (n-r)\ r\text{'s)} \\ &> (C\cdot r^r)\cdot r\cdot\cdots\cdot r\cdot r && \text{by choice of } n \\ &= C\cdot r^n. \end{aligned}$$

□

Question 4.4. Theorem 4.2 says that $n! \neq O(2^n)$. Find an integer N so that for $n \geq N$, $n! > 10 \cdot 2^n$.

We state without proof a result that gives some feeling for the size of $n!$.

Stirling's Formula. $n!$ is approximately equal to the function

$$\sqrt{2\pi n}\left(\frac{n}{e}\right)^n$$

where $e = 2.71828\ldots$ is the base of the natural logarithm.

We do not define what "approximately equal" means, but a consequence of Stirling's formula is that both

$$n! = O\left(\left(\frac{n}{e}\right)^n \sqrt{n}\right) \quad \text{and} \quad \left(\frac{n}{e}\right)^n \sqrt{n} = O(n!).$$

By all accounts, $n!$ quickly becomes humongous.

We have used the word permutation previously in Chapter 1 in the study of functions. We said that a one-to-one and onto function f, $f:D \to D$, is called a permutation. When D is a finite set, such as $I_n = \{1, 2, \ldots, n\}$, then the two definitions are really the same. If $f:I_n \to I_n$ is a one-to-one function (which is necessarily onto by Theorem 9.1 from Chapter 1), then we get a permutation, in the sense of this section, from the listing $\langle f(1)\ f(2)\ \ldots\ f(n)\rangle$. Conversely, given a permutation of I_n $\langle a_1\ a_2\ \ldots,\ a_n\rangle$, this defines a one-to-one function $f:I_n \to I_n$ by setting $f(i) = a_i$ for $i = 1, \ldots, n$.

Example 4.5. Corresponding to the permutation $\langle 1\ 3\ 2\ 5\ 4\ 6\rangle$ is the one-to-one function f defined by $f(1) = 1$, $f(2) = 3$, $f(3) = 2$, $f(4) = 5$, $f(5) = 4$, and $f(6) = 6$. Corresponding to the one-to-one function f defined by $f(i) = 7 - i$ for $i = 1, 2, \ldots, 6$, is the permutation $\langle 6\ 5\ 4\ 3\ 2\ 1\rangle$.

Permutations occur in many mathematical and computer science contexts. In this course we shall meet them again in a game in the next section as well as in later chapters. In other courses they occur in diverse settings such as linear algebra, sorting and searching problems, in a variety of games and path traversing questions, and even in English bell ringing. The goal in English bell ringing, called change ringing, is to ring all permutations on however many bells are in the tower. Fortunately, most towers have at most 8 or 10 bells.

EXERCISES FOR SECTION 4

1. List all permutations of $\{x, y, z\}$ and of $\{a, b, c, d\}$.
2. Find the least integer n such that $n! > 3^n$. Then find any integer n such that $n! > 4^n$.
3. English bell ringers, called change ringers, ring tower and hand bells following sequences of permutations; however, one of the requirements of change ringing is that a permutation p may be followed by a permutation p' only if for $i = 2, 3, \ldots, n-1$, the number in position i of p is in position $i-1$, i or $i+1$ of p'. The number in position 1 of p may stay in position 1 or move to position 2, and the number in position n may stay in position n or move to position $n-1$ of p'. Thus the following sequence of three permutations is legal from

the perspective of bell ringing: $\langle 1\ 2\ 3\rangle$, $\langle 1\ 3\ 2\rangle$, $\langle 3\ 1\ 2\rangle$, but the next permutation (as produced by PERM) $\langle 2\ 1\ 3\rangle$ is not a legal follow-up to $\langle 3\ 1\ 2\rangle$, since number 3 jumped from the front to the back. Find a sequence of all 3! permutation on $\{1, 2, 3\}$ that is suitable for bell ringing. Do the same for the 4! permutations on $\{1, 2, 3, 4\}$.

4. Which is larger $n^{\sqrt{n}}$ or $(\sqrt{n})^n$?

5. Prove that $n^{\sqrt{n}} = O(2^n)$. Either prove or disprove the statement $(\sqrt{n})^n = O(2^n)$.

6. Prove that $n^{\sqrt{n}} \neq O(p(n))$ for any polynomial $p(n)$.

7. Prove that $r^n = O(n!)$.

8. Prove the following corollary of Theorem 4.2: For every real number r, $n! \neq O(r^n)$.

9. Check the accuracy of Stirling's formula: For $n = 5, 6, \ldots, 10$ use a calculator to evaluate $\sqrt{2\pi n}(n/e)^n$ using $e = 2.71828$ and compare these values with $n!$.

10. Let f be the one-to-one function defined on $\{1, 2, \ldots, 6\}$ by $f(1) = 2$, $f(2) = 1$, $f(3) = 4$, $f(4) = 3$, $f(5) = 6$, and $f(6) = 5$. Show that $f^2 = f \circ f$ is the identity mapping. Find a one-to-one function g on the same domain for which g^2 is not the identity. For this g find the least integer i such that g^i (g composed with itself i times) is the identity. Then write down the i permutations: $\langle g(1) \ldots g(6)\rangle$, $\langle (g(1))^2 \ldots (g(6))^2\rangle, \ldots$, and $\langle (g(1))^i \ldots (g(6))^i\rangle$. Are these all of the possible permutations on six numbers?

11. Given a permutation $p = \langle a_1\ a_2\ \ldots\ a_n\rangle$, an **adjacent transposition** is the switching of two items in positions i and $(i + 1)$ for some value of $i = 1, 2, \ldots, n - 1$. For example, $\langle a_2\ a_1\ \ldots\ a_n\rangle$ is the result of applying an adjacent transposition to p as is $\langle a_1\ a_2\ \ldots\ a_n\ a_{n-1}\rangle$. Given permutations p and p' we define the distance between them to be the minimum number of adjacent transpositions needed to transform one into the other; we denote this by $d(p, p')$. Determine the following values of $d(p, p')$:
   (a) $p = \langle 1\ 2\ 3\ 4\rangle$ $\quad p' = \langle 4\ 3\ 2\ 1\rangle$
   (b) $p = \langle 1\ 2\ 3\ 4\rangle$ $\quad p' = \langle 3\ 1\ 4\ 2\rangle$
   (c) $p = \langle 3\ 2\ 1\ 4\rangle$ $\quad p' = \langle 1\ 3\ 2\ 4\rangle$.

12. We define a permutation p to be either **even** or **odd** according as $d(p, p^*)$ is an even or an odd number when p^* is the identity permutation $\langle 1\ 2\ 3\ \ldots\ n\rangle$. Determine which of the four nonidentity permutations of the previous problem are even and which are odd.

13. For which permutation p on $\{1, 2, \ldots, n\}$ is $d(p, p^*)$ the largest when p^* is the identity permutation? Also, what is the value of $d(p, p^*)$ for this permutation?

14. How many even and how many odd permutations are there on $\{1, 2, \ldots, n\}$? Prove your answer.

15. Modify the algorithm PERM so that it outputs only even permutations.

16. Design an algorithm that upon input n, a positive integer, lists all permutations of $\{1, 2, \ldots, n\}$ without first listing all permutations on $\{1, 2, \ldots, n-1\}$. (*Hint:* Look at the order of the output of PERM and see if there is a way to move directly from one permutation to the next.)
17. Prove that $n! \leq n^n$ and thus show that n^n is a new candidate for the biggest function in the hierarchy.
18. Find a function $f(n)$ such that $f(n) \neq O(n^n)$.
19. Let P be the positive integers and suppose that we define a permutation of P to be a reordering of P. If f is a function with domain and target P, what properties must f have so that a listing of its range $\langle f(1)\ f(2) \ldots \rangle$ is a permutation (or reordering) of P?
20. There are $n!$ different ways to make a straight line of n people. Suppose instead that n people are seated at a circular table and two seating arrangements are considered the same if everyone has the same person on their left side and the same person on their right side. How many different seating arrangements are there? Suppose that two arrangements are considered the same if everyone has the same set of two people on their left and right, but which side doesn't matter. Then how many different seating arrangements are there?

3:5 AN APPLICATION OF PERMUTATIONS: THE GAME OF MASTERMIND

We turn to some colorful "near" permutations that arise in the game of Mastermind. Mastermind is a two-person guessing game that is usually played without special knowledge of permutations and set theory. As you read this section, you might consider whether mathematical training helps in playing this game.

Mastermind is played as follows. The first player secretly writes down a list or code of four colors, chosen from red, yellow, green, blue, purple, and white. We denote the set of colors by

$$C = \{r, y, g, b, p, w\}$$

where each initial stands for the color beginning with the same letter. Repetitions of colors in the secret list are allowed. Then the second player tries to guess the colors and their order in the secret list.

For example, suppose that I choose

$$p\ b\ p\ g$$

as my secret list. Then when you guess a list of four colors, I must tell you two pieces of information. First I must tell you how many of your colors are correct and in the right position. Next I compare the remaining colors in my secret code

Table 3.3

Guess	*Position and Color Correct*	*Color Correct: Position Incorrect*
r b r b	1	0
b g b g	1	1

and in your guess and tell you how many of these colors match correctly but are in the wrong position. Two guesses and responses are listed in Table 3.3.

To use the language of set theory, both the secret code and all guesses come from the Mastermind universe, which is the Cartesian product C^4. When player 2 makes a guess, that is, selects an element from C^4, player 1 produces f(guess), where f is a function whose domain is C^4 and whose target is $\{0, 1, 2, 3, 4\}^2$. Thus the image of any particular guess is an ordered pair of integers as described above.

Example 5.1. The answers to some possible guesses of a secret code are listed in Table 3.4.

Table 3.4

Guess	*Position and Color Correct*	*Color Correct: Position Incorrect*
r r r r	1	0
y y y y	1	0
g g g g	0	0
b b b b	1	0
p p p p	0	0
w w w w	1	0

Question 5.1. Figure out the four colors involved in the secret code of Example 5.1. Can you tell what their order is?

Example 5.1 (continued). Some more guesses and answers about the secret code in Example 5.1 are shown in Table 3.5.

Table 3.5

r y b w	1	3
r y w b	0	4
r w y b	1	3
w r y b	0	4

Question 5.2. Can you determine from the information in Example 5.1 what the secret code is? If not, try to figure out as many of the correct color positions as possible from these answers.

Question 5.3. Find a classmate, roommate, or any willing soul, and play six games of Mastermind in three of which you pick the secret code and in three of which you try to guess the secret code. How many guesses were needed in each game?

Experienced Mastermind players can figure out the secret code in four or five guesses. Their strategy combines a knowledge of good initial guesses plus close logical analysis of the responses to these. We develop some strategies of our own now. One strategy would be to repeatedly pick elements of C^4 at random until an answer of (4, 0) results. Since there are six colors and four positions to fill with repetitions possible, there are $6^4 = 1296$ different possible Mastermind codes. Thus random guessing will not be an effective playing strategy! We shall describe a method that can determine the list of colors with only six guesses.

An effective playing strategy for this game (and any game) is just an algorithm that tells us what to do in any possible situation. We shall develop three different strategies for Mastermind. We begin with one that comes from the idea behind the guesses used in Example 5.1.

Problem. To determine the secret list of colors in a Mastermined game.

Algorithm 1

STEP 1. For each color x in $\{r, y, g, b, p, w\}$ guess $x\ x\ x\ x$.

STEP 2. (Now that you know the colors in the list) guess all possible lists using the four (or fewer) colors involved.

Notice that both steps 1 and 2 really involve several steps. Step 1 is shorthand for six steps or six guesses.

Question 5.4. Suppose that you learn in step 1 that the colors involved are green, blue, purple, and white. How many different secret codes are there involving these colors? How could you go about writing down all of these possible lists?

In fact, when we discover that the code involves four different colors, then the code is one of the permutations of these colors. But what happens when there are repeated colors?

Example 5.2. Suppose that the colors involved in the secret code are green, blue, and purple and that green occurs twice. Then the code will be a "**near**" **permutation** of $\{g, g, b, p\}$, that is, an ordered list of these letters including two g's. (A "real"

permutation is an ordered list of distinct elements.) A set with repeated elements is known as a **multiset**. However, our counting and algorithmic techniques will solve this case too. In each code word the g's will occupy two positions, for example, the first and the third position. We associate their positions with the 2-subset (in this instance $\{1,3\}$) of the four possible positions $P = \{1,2,3,4\}$. Conversely, every 2-subset of P gives us a prescription for where to place the g's. Thus there are $\binom{4}{2}$ ways to fill in the g's into a color code. The remaining two places can be filled with first b and then p or with first p, then b. Thus the total number of codes using $\{g,\ g,\ b,\ p\}$ is $\binom{4}{2} \cdot 2 = 12$.

Furthermore, PAIR and JSET, with $n = 4$ and $j = 2$, provide a list of all 2-subsets of the 4-set P, and from these we position the g's. Then each of these leads to two code words by filling in b and g in the two possible orders. For example $\{1,2\}$ leads to $g\ g\ b\ p$ and $g\ g\ p\ b$.

Question 5.5. List all possible codes formed from $\{g,\ g,\ b,\ p\}$.

Question 5.6. Look back at Algorithm 1. What is the maximum number of steps or guesses that you will need in step 2?

The total number of guesses needed in Algorithm 1 is rather large, and you probably can think of better ways to play. Here is another approach that may more closely resemble how you and most people play Mastermind. The idea is to focus on the set of all possible codes, then on a subset of this set, and on a subset of the subset until the subset is pared down to one code.

Algorithm 2

STEP 1. Let L = {all possible Mastermind codes}
STEP 2. Repeat
Begin
STEP 3. Pick an element from L to be your guess
STEP 4. Update L {remove from L all those codes that are inconsistent with the response to your guess}
End
Until your guess is correct
STEP 5. Stop.

COMMENT. The form of step 2, "**Repeat** . . . **Until** . . ." is like the While . . . do loop, only the condition for ending the loop is checked at the end of the execution of the loop.

Algorithm 2 requires a lot of work. Initially, the list L in step 1 has 1296 codes in it, and the checking and updating will be time-consuming. Neither this approach nor that of Algorithm 1 would be so bad for a computer to implement, since we can use the computer's large memory and quick access to it to play the game effectively.

Here is a third approach, possibly effective for use by both humans and computers.

Algorithm 3

STEP 1. Make the following four guesses:

$$r\ r\ y\ y$$
$$r\ g\ r\ g$$
$$b\ b\ p\ p$$
$$b\ w\ b\ w$$

STEP 2. Create L, a list of all possible Mastermind codes that are consistent with the answers you receive from the first four guesses

STEP 3. Repeat
Begin
STEP 4. Pick an element from L to be your guess
STEP 5. Update L {remove from L all those codes that are inconsistent with the response to your guess}
End
Until your guess is correct

STEP 6. Stop.

Try playing a few games using this algorithm. You will be surprised at how small the list L is that you create in step 2. In fact, we claim that after at most one more guess you can always figure out the secret list; so, in a total of six guesses you will have the answer nailed down. (See Supplementary Exercise 15.)

There are a few ideas that can be derived from thinking about and playing Mastermind. At first you might wonder why we bother to create a formal algorithm to play the game. We all can quickly learn effective guesses to make, and so why not play just using these hunches and logical deductions? In fact, that's how most of us do play this and many other games. But as soon as we turn the process around so that the computer is making the guesses, then it becomes essential to have an algorithm so that the computer has a way to proceed. If we start to analyze how we play the game, we soon see that we have lots of different strategies that we adapt according to the responses. It would be quite a feat to design an algorithm that would accommodate all the different possibilities that the mind thinks up. In fact, it is quite impressive how clever and logical the human mind is.

Instead of trying to simulate the logical working of a human brain, we have chosen very straightforward algorithms that could be programmed easily and that will lead to success with a relatively small number of guesses. We have also exploited the fact that the computer can do lots of checking rather quickly. You are invited to write (or find a friendly computer programmer to write) a program using Algorithms 2 or 3, and then to race the computer to see who is faster!

EXERCISES FOR SECTION 5

1. Find a solution to each of the following Mastermind games given the indicated guesses and answers:

	Guess	*Position and Color Correct*	*Color Correct; Position Incorrect*
(a)	*r b r b*	0	0
	w y w y	0	0
	g p g p	3	0
	g g p g	0	2
(b)	*r b r b*	2	0
	w w y y	0	2
	r b y w	1	3
	y b r w	0	4
(c)	*r y w b*	1	1
	g b p w	1	0
	r y p w	2	0
	y g b p	0	0
	w r w r	1	2
(d)	*r r y y*	0	0
	r g r g	0	0
	b b p p	2	0
	b w b w	1	1
	w p w p	2	2
(e)	*b g p r*	0	4
	r p g b	2	2
	r g p b	1	3
	r p b g	1	3
	r b g p	0	4

2. Explain why there are no solutions to the following guesses and answers:

(a)	*r b r b*	0	1
	y g y g	1	0
	p w p w	0	1
	r y p w	1	3

(*b*)	*r g b b*	2	2
	b g r b	1	3
	g r b b	0	4
(*c*)	*b r g b*	0	3
	y y p w	1	0
	y r g b	1	2
	r b y b	2	1

3. Suppose you learn that there are two colors, green and blue, in the code word. How many code words are possible? List them all.
4. Write a list of four Mastermind guesses and responses for which there are **(a)** no solutions, **(b)** one solution, and **(c)** two or more solutions.
5. How many Mastermind codes begin with r? How many Mastermind codes begin with r and contain no other r? How many Mastermind codes begin with r and contain another r?
6. How many possible codes are there for which the response to the guess $r\ b\ b\ b$ is 0 (color and position correct) and 1 (color but not position correct)?
7. List all Mastermind codes formed from $\{p, p, y, b\}$. Then do the same for $\{p, p, y, y\}$.
8. Associated with the game of Mastermind is a function $f: C^4 \to \{0, 1, 2, 3, 4\}^2$ as described in this section. What is the range of f?
9. Answer the following questions (without listing all possibilities.) Suppose that we play a version of Mastermind in which we pick a code of length 5 from the same set of six colors. Otherwise, all the rules are the same.
   (*a*) How many possible code words are there?
   (*b*) How many code words are there in which the five colors $\{r, b, g, y, p\}$ each appear once?
   (*c*) How many code words are there if the colors $\{r, b, g\}$ each appear once and w appears twice?
   (*d*) How many code words are there if the colors $\{r, b\}$ each appear once and p appears three times?
10. How many code words are consistent with the following set of guesses and responses?

r r y y	1	0
r g r g	0	0
b b p p	1	0
b w b w	2	1

11. Define a new game called Trivialmind in which only three colors are used (red, blue, and white) and the secret code consists of a list of only two colors. How many different secret color codes are there in Trivialmind?

12. Describe a variation on Algorithm 1 that will work for Trivialmind. What is the maximum number of guesses that will be made using this algorithm?

13. Describe a variation on Algorithm 3 that will work for Trivialmind. Using this new version of Algorithm 3, how many guesses will player 2 need to guarantee a correct guess?

14. Show that no algorithm will in all cases correctly solve Trivialmind with three or fewer guesses.

15. The point of this exercise is to improve Algorithm 1. In step 1 we guessed $x\ x\ x\ x$ for each x in $\{r, y, g, b, p, w\}$. Figure out a way to reduce these six questions to five. Now suppose that we know the colors present in the secret code and their frequency. Suppose that the color x is present (at least once) and that the color z is missing. Notice that at least two colors must be missing. What can you learn from the following four questions:

$$z\ z\ z\ x$$
$$z\ z\ x\ z$$
$$z\ x\ z\ z$$
$$x\ z\ z\ z?$$

Can you learn this same information with only three guesses? Using these ideas, try to devise an algorithm along these lines with as few guesses as possible. How many guesses do you make in general?

16. Pick a mastermind code and apply Algorithm 3. How large is your initial list L?

17. Suppose that you guess $r\ b\ r\ b$. How many different responses are possible? If you guess $r\ b\ r\ w$, how many different responses are there? What is the minimum number of different responses to any Mastermind guess? What are the guesses that produce this minimum? What is the maximum number of different responses to any Mastermind guess? What are the guesses that produce this maximum?

3:6 THE BINOMIAL THEOREM

This chapter concludes with a result that unifies a variety of facts about binomial coefficients. We know, for example, that the sum of the binomial coefficients $\binom{n}{i}$ for $i = 0, 1, \ldots, n$ equals 2^n, and from Exercise 3.10 we know that the sum of the "even" binomial coefficients $\binom{n}{2j}$ for $j = 0, 1, \ldots, \lfloor n/2 \rfloor$ equals 2^{n-1}. There are many other similar identities lurking around, waiting to be discovered.

Question 6.1. For $n = 3, 4$, and 5 verify that

$$\binom{n}{0} - \binom{n}{1} + \binom{n}{2} - \binom{n}{3} + \cdots + (-1)^i \binom{n}{i} + \cdots + (-1)^n \binom{n}{n} = 0.$$

Question 6.2. Verify that

$$\binom{2}{2} + \binom{3}{2} + \binom{4}{2} = \binom{5}{3}.$$

In general, the sum of the binomial coefficients

$$\binom{2}{2} + \binom{3}{2} + \cdots + \binom{i}{2} + \cdots + \binom{n}{2}$$

can be expressed as a single binomial coefficient $\binom{k}{j}$. For $n = 5$ and 6 find such a binomial coefficient $\binom{k}{j}$. Then for arbitrary n find k and j so that

$$\binom{2}{2} + \binom{3}{2} + \cdots + \binom{i}{2} + \cdots + \binom{n}{2} = \binom{k}{j}.$$

(Here j will be a constant and k will depend upon n.)

Example 6.1. We review some polynomial arithmetic:

$$(1 + x)^2 = 1 + 2x + x^2$$
$$(1 + x)^3 = 1 + 3x + 3x^2 + x^3$$
$$(1 + x)^4 = 1 + 4x + 6x^2 + 4x^3 + x^4.$$

Notice that the coefficients in the expansion above of $(1 + x)^k$ are exactly the numbers in the kth row of Pascal's triangle. Thus the coefficient of x^i in $(1 + x)^n$ appears to be the binomial coefficient $\binom{n}{i}$ for $0 \leq i \leq n$.

Example 6.1 sets us on the right track. We prove the suggested result and find that the answers to Questions 6.1 and 6.2 and much more follow from the so-called Binomial Theorem.

Theorem 6.1. The Binomial Theorem. For n, a natural number,

$$(1 + x)^n = \binom{n}{0} + \binom{n}{1}x + \binom{n}{2}x^2 + \cdots + \binom{n}{i}x^i + \cdots + \binom{n}{n}x^n.$$

Proof. We prove this theorem by induction on n. When $n = 0$,

$$(1 + x)^0 = 1 = \binom{0}{0}.$$

When $n = 1$,

$$(1 + x)^1 = (1 + x) = \binom{1}{0} + \binom{1}{1}x.$$

(You might check the cases of $n = 2$ and 3 to make sure that the statement of the theorem is clear.)

The inductive hypothesis is that

$$(1 + x)^k = \binom{k}{0} + \binom{k}{1}x + \cdots + \binom{k}{i}x^i + \cdots + \binom{k}{k}x^k.$$

From this we must determine $(1 + x)^{k+1}$. By factoring,

$$\begin{aligned}(1 + x)^{k+1} &= (1 + x)^k(1 + x) \\ &= \left[\binom{k}{0} + \binom{k}{1}x + \cdots + \binom{k}{i}x^i + \cdots + \binom{k}{k}x^k\right](1 + x)\end{aligned}$$

by the inductive hypothesis,

$$\begin{aligned}&= \left[\binom{k}{0} + \binom{k}{1}x + \cdots + \binom{k}{i}x^i + \cdots + \binom{k}{k}x^k\right] \\ &\quad + \left[\binom{k}{0}x + \cdots + \binom{k}{i-1}x^i + \cdots + \binom{k}{k-1}x^k + \binom{k}{k}x^{k+1}\right]\end{aligned}$$

by multiplying first by 1 and then by x.

Now we must collect like terms together. That is, first we find the constant term, the coefficient of x^0. It is just $\binom{k}{0} = 1 = \binom{k+1}{0}$, since all terms in the second bracket involve x. What are the terms involving $x^1 = x$? We have $\binom{k}{1}x$ and $\binom{k}{0}x$, from the first and second bracket, respectively, or

$$\binom{k}{1}x + \binom{k}{0}x = (k + 1)x = \binom{k+1}{1}x.$$

In general, what are the coefficients of x^i? In the first bracket we have $\binom{k}{i}x^i$ and in the second bracket we have $\binom{k}{i-1}x^i$. Thus in the sum we have

$$\binom{k}{i}x^i + \binom{k}{i-1}x^i = \left[\binom{k}{i} + \binom{k}{i-1}\right]x^i$$
$$= \binom{k+1}{i}x^i \qquad \text{by Corollary 2.2.}$$

Finally, the term x^{k+1} appears only in the second bracket and we have

$$\binom{k}{k}x^{k+1} = x^{k+1} = \binom{k+1}{k+1}x^{k+1}. \qquad \square$$

For comparison we offer the following.

Second proof of Theorem 6.1. We want to determine the expansion of $(1 + x)^n$ as a polynomial:

$$(1 + x)^n = a_0 + a_1x + \cdots + a_ix^i + \cdots + a_nx^n.$$

That is, we want to determine the coefficients $a_0, a_1, \ldots, a_n$. Now

$$(1 + x)^n = (1 + x)(1 + x)\cdots(1 + x),$$

a product of n identical factors. A term in the product results from every way of selecting one element (either 1 or x) from each pair of parentheses and multiplying them together. For example, if we select 1 from the first parenthesis, x from the second and third, and 1 from all the rest, we multiply these to get the product x^2. Thus this selection contributes 1 to a_2, the coefficient of x^2. Suppose that we designate this choice by an n-vector with a zero in the entries where we select a $1 (= x^0)$, and a one in the entries where we select an $x (= x^1)$. Thus in our example we designate our selection by the vector $(0, 1, 1, 0, 0, \ldots, 0)$. Then the number of ways to choose elements one from each parenthesis, to multiply together and get x^2 is the same as the number of bit vectors with two ones and $(n - 2)$ zeros. From Example 3.2 we know that the number of such vectors equals $\binom{n}{2}$, and so $a_2 = \binom{n}{2}$.

In general, a_i, the coefficient of x^i in the expansion of $(1 + x)^n$ equals the number of ways to select i xs and $(n - i)$ 1s, one from each of the n sets of parentheses. This number equals the number of bit vectors with i ones and $(n - i)$ zeros or $\binom{n}{i}$ as we saw in Example 3.2. $\square$

Notice that Theorem 6.1 gives us a polynomial identity, that is, an equation that is true upon substitution of any real value of x. We now substitute different values for x and see what happens.

Example 6.2. Substituting $x = 1$ in the Binomial Theorem shows that

$$\begin{aligned} 2^n &= (1+1)^n \\ &= \binom{n}{0} + \binom{n}{1}1 + \binom{n}{2}1^2 + \cdots + \binom{n}{n}1^n \\ &= \binom{n}{0} + \binom{n}{1} + \cdots + \binom{n}{n}, \end{aligned}$$

the same result as in Corollary 3.2.

Example 6.3. Substituting $x = -1$ in the Binomial Theorem leads to

$$\begin{aligned} 0 &= (1-1)^n \qquad \text{if } n > 0 \\ &= \binom{n}{0} + \binom{n}{1}(-1) + \binom{n}{2}(-1)^2 + \cdots + \binom{n}{n}(-1)^n \\ &= \binom{n}{0} - \binom{n}{1} + \binom{n}{2} - \binom{n}{3} + \cdots + (-1)^n\binom{n}{n}, \end{aligned}$$

a phenomenon we observed in Question 6.1. Notice that we may rearrange the last equation to read

$$\binom{n}{0} + \binom{n}{2} + \binom{n}{4} + \cdots = \binom{n}{1} + \binom{n}{3} + \binom{n}{5} + \cdots.$$

Since the sum of the left-hand side plus the right-hand side equals 2^n, we see that each side separately equals $(\frac{1}{2})(2^n) = 2^{n-1}$, a result that we noted in the first paragraph of this section.

How does Question 6.2 relate to the Binomial Theorem? We can't obtain the result suggested there by merely substituting in a number for x, but we can use the Binomial Theorem to derive it. We have previously seen in Chapter 2, Example 3.3, another polynomial identity that gives the sum of a (finite) geometric series:

$$1 + x + x^2 + \cdots + x^i + \cdots + x^n = \frac{1 - x^{n+1}}{1 - x}.$$

Since this is an identity that holds true for all x, except for $x = 1$, we can substitute both values and expressions for x, and the equation still holds true. For example,

if we substitute $(1 + x)$ for x everywhere in the geometric series we get the result that

$$\begin{aligned} &1 + (1 + x) + (1 + x)^2 + \cdots + (1 + x)^i + \cdots + (1 + x)^n \\ &\quad = \frac{1 - (1 + x)^{n+1}}{1 - (1 + x)} \qquad \text{provided } x \neq 0 \\ &\quad = \frac{1 - (1 + x)^{n+1}}{-x} \\ &\quad = \frac{(1 + x)^{n+1} - 1}{x} \qquad \text{for } x \neq 0. \end{aligned} \tag{*}$$

This result tells us that if we multiply out both the right-hand side and the left-hand side of equation (*), then the resulting polynomials are the same.

In particular, let's look at the x^2 term on both sides of equation (*). On the left-hand side at first there are no x^2 terms, but then x^2 appears in the expansion of all the terms from $(1 + x)^2$ on up to $(1 + x)^n$. By the Binomial Theorem, here are the x^2 terms:

$$\begin{aligned} &\binom{2}{2}x^2 + \binom{3}{2}x^2 + \cdots + \binom{i}{2}x^2 + \cdots + \binom{n}{2}x^2 \\ &\quad = \left[\binom{2}{2} + \binom{3}{2} + \cdots + \binom{i}{2} + \cdots + \binom{n}{2}\right]x^2. \end{aligned}$$

The x^2 term on the right-hand side of (*) will result from dividing the x^3 term of the numerator by the x in the denominator. By the Binomial Theorem the x^3 term of $[(1 + x)^{n+1} - 1]$ is $\binom{n+1}{3}x^3$. Thus

$$\binom{n+1}{3}\frac{x^3}{x} = \binom{n+1}{3}x^2.$$

Since (*) gives two expressions for the same underlying polynomial, the coefficient of x^2 on the left and on the right must be the same, that is,

$$\left[\binom{2}{2} + \binom{3}{2} + \cdots + \binom{i}{2} + \cdots + \binom{n}{2}\right]x^2 = \binom{n+1}{3}x^2,$$

or

$$\binom{2}{2} + \binom{3}{2} + \cdots + \binom{i}{2} + \cdots + \binom{n}{2} = \binom{n+1}{3}.$$

There is another more general form of the Binomial Theorem that gives the expansion of a polynomial with two variables:

$$(y + x)^n = \binom{n}{0} y^n x^0 + \binom{n}{1} y^{n-1} x^1 + \cdots + \binom{n}{i} y^{n-i} x^i + \cdots + \binom{n}{n} y^0 x^n.$$

This result can be proved in essentially the same way as Theorem 6.1.

A common problem in combinatorial mathematics is to count objects with specified properties, for example, to count the number of j-subsets of an n-set. Sometimes a neat formula can be obtained; other times the answer can be derived from a polynomial identity. One often can deduce that the answer to the problem is, say, the coefficient of x^j in a certain polynomial. We have seen that the number of j-subsets of an n-set is the coefficient of x^j in the polynomial $(1 + x)^n$. Such polynomial solutions are known as **generating functions**. The use of generating functions is a powerful technique with wide application. In this section we have seen an introduction to the methods and use of generating functions.

EXERCISES FOR SECTION 6

1. Expand $(1 + x)^5$ and $(1 + x)^6$ and check that the coefficients are the binomial coefficients promised by Theorem 6.1. [Reminder: Do you really have to multiply $(1 + x)$ by itself four and five times, respectively?]
2. ***(a)*** Find a simple formula for

$$\binom{n}{0} + \binom{n}{1} 2 + \binom{n}{2} 4 + \cdots + \binom{n}{i} 2^i + \cdots + \binom{n}{n} 2^n,$$

and for

$$\binom{n}{0} - \binom{n}{1} 2 + \binom{n}{2} 4 - \cdots + (-1)^i \binom{n}{i} 2^i + \cdots + (-1)^n \binom{n}{n} 2^n.$$

(b) Prove that if n is even, then

$$\binom{n}{0} + \binom{n}{2} 4 + \binom{n}{4} 16 + \cdots + \binom{n}{n} 4^{n/2} = \frac{3^n + 1}{2}.$$

3. Use the Binomial Theorem to derive the expansion of $(1 + 1/x)^n$.
4. In each of the following, first verify the equation by selecting a value for r, n, and m and checking the equation for these values. Then prove the general result about sums of binomial coefficients.

(*a*) $\binom{r}{0}+\binom{r+1}{1}+\cdots+\binom{r+i}{i}+\cdots+\binom{r+n}{n}=\binom{r+n+1}{n}$.

(*b*) For $m \le n$, $\binom{m}{m}+\binom{m+1}{m}+\cdots+\binom{i}{m}+\cdots+\binom{n}{m}=\binom{n+1}{m+1}$.

(*c*) For $n < r$, $\binom{r}{0}-\binom{r}{1}+\cdots+(-1)^i\binom{r}{i}+\cdots+(-1)^n\binom{r}{n}=(-1)^n\binom{r-1}{n}$.

5. (*a*) For $r \le 5$, verify that

$$\binom{5}{0}\binom{6}{r}+\binom{5}{1}\binom{6}{r-1}+\binom{5}{2}\binom{6}{r-2}+\cdots+\binom{5}{r}\binom{6}{0}$$
$$=\binom{5+6}{r}=\binom{11}{r}.$$

(*b*) Now we want to show, in general, that for r less than or equal to both m and n,

$$\binom{m}{0}\binom{n}{r}+\binom{m}{1}\binom{n}{r-1}+\binom{m}{2}\binom{n}{r-2}+\cdots+\binom{m}{r}\binom{n}{0}=\binom{m+n}{r}.$$

It is clear that $(1+x)^m(1+x)^n=(1+x)^{m+n}$. Use the Binomial Theorem to find the coefficient of x^r in $(1+x)^{m+n}$. Then use the Binomial Theorem to expand $(1+x)^m$ and $(1+x)^n$ and determine the coefficient of x^r in the product

$$(1+x)^m(1+x)^n.$$

From these results, conclude that the previous equation holds.

6. Explain why

$$(1+x)^n\left(1+\frac{1}{x}\right)^n=\frac{(1+x)^{2n}}{x^n}.$$

Then find the constant term (the coefficient of x^0) in $(1+x)^n\left(1+\frac{1}{x}\right)^n$ and find the constant term in the expansion of $\frac{(1+x)^{2n}}{x^n}$. See Supplementary Exercise 17 for an application.

7. Prove the general form of the Binomial Theorem that gives the expansion of $(y+x)^n$.

8. Find a simple formula for the sum of the binomial coefficients

$$\binom{3}{3} + \binom{4}{3} + \cdots + \binom{i}{3} + \cdots + \binom{n}{3}.$$

Next prove the formula using the ideas of the solution to Question 6.2. Then find and prove a formula for the sum

$$\binom{k}{k} + \binom{k+1}{k} + \cdots + \binom{i}{k} + \cdots + \binom{n}{k} \qquad \text{for } k \leq n.$$

3:7 IMPORTANT SUBSETS

The most important ideas in this chapter are those about sets: j-subsets, permutations, and the counting of these sets. It is worth repeating that this set theory along with that in Chapter 1 is a crucial element in all further study of mathematics, computer science, and algorithms. The new counting functions, the factorial and the binomial coefficients, will appear frequently. Already the latter have given us the number of j-subsets of an n-set, the number of shortest paths in a rectangular grid, the number of n-bit sequences with a specified number of ones, and the coefficients in the expansion of $(1 + x)^n$. We shall meet these functions in every future chapter; we shall use the subset and permutation ideas repeatedly and without further ado. To be honest, Mailmobiles haven't caught on with quite the popularity initially expected, but the path counting problem is a classic.

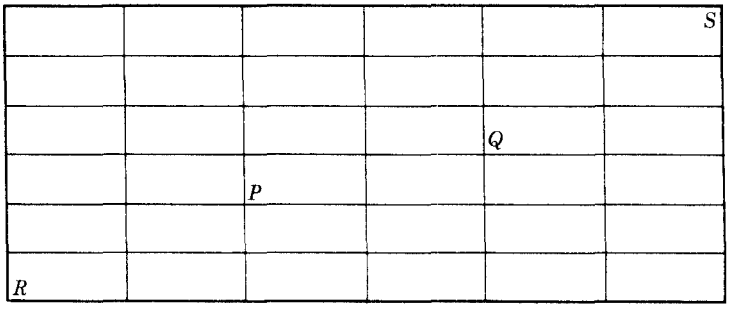
The algorithms of this chapter, JSET and PERM, are of more than illustrative value. They are used repeatedly, for example, in Mastermind and in more serious applications in Chapter 5. In these algorithms we introduced nested loops and Boolean variables. Although these algorithms and algorithmic techniques are more complex than those of the previous chapter, they were selected for their relative simplicity. The "state of the art" algorithms for generating subsets and permutations are more sophisticated and a little more efficient. However, no amount of trickery can get around the fact that generating all permutations of an n-set and all subsets of size $\lfloor n/2 \rfloor$ requires an exponential amount of work.

This study of counting, generating, and listing objects of a certain size provides an introduction to the areas of combinatorial analysis and combinatorial algorithms. In these modern and fast-growing fields, a typical problem involves counting all objects with a particular structure and listing them in a certain order. The counting task is often solved using functions like the binomial coefficients and techniques like generating functions. The listing is frequently done by imposing an order on the structure such as lexicographic order.

In Chapter 5 we shall focus on finite sets and their 2-subsets, an area known as graph theory. The study of orderings, both constructing them and searching them, is the subject of Chapter 6.

SUPPLEMENTARY EXERCISES FOR CHAPTER 3

1. Given the rectangular grid shown here how many different shortest paths join the vertices labeled R and S? How many different shortest paths are there from R to S that do not go through the vertex labeled P? How many shortest paths are there from R to S that do not go through either the vertex labeled P or the vertex labeled Q?



2. How many 4-subsets of $\{1, \ldots, 8\}$ contain the number 3? How many contain 5? How many 4-subsets contain either 3 or 5?
3. Count the number of 10-digit ternary numbers (i.e., numbers that use only the digits 0, 1, and 2). How many of these contain exactly four zeros? How many contain exactly four ones? How many contain three zeros and three ones?
4. Find a formula for the number of n-digit ternary numbers that contain i zeros and $(n - i)$ ones and twos (combined). How many contain exactly i zeros and j ones when i and j are positive integers with $i + j \leq n$?
5. The number of n-letter "words" made up from the letters a, b, and c is the same as the number of n-digit ternary numbers—why? How many n-letter words are there with i a's, j b's and $(n - i - j)$ c's?
6. **(a)** Is the following true or false? $(2n)! = 2^n \cdot n!$ Explain.
   (b) For an odd number $(2n - 1)$ we define the odd factorial

$$(2n - 1)O! = (2n - 1)(2n - 3) \cdots \cdot 3 \cdot 1.$$

Thus $(3)O! = 3 \cdot 1 = 3$, and $(5)O! = 5 \cdot 3 \cdot 1 = 15$. Calculate $(7)O!$ and $(9)O!$.

(*c*) Explain why the following new formula is correct:

$$\binom{2n}{n} = \frac{2^n \cdot (2n-1)O!}{n!}.$$

(*d*) Show that $(2n-1)O!/n! < 2n$ and then deduce that $\binom{2n}{n} = O(4^n)$.

7. Define the even factorial $(2n)E!$ in a way analogous to the odd factorial $(2n-1)O!$. Then find a formula that relates $(2n)E!$ and $n!$.

8. Reread Exercise 2.12 where the factorial representation of a number is defined. Then prove that every positive integer has a unique factorial representation.

9. Find an algorithm that with input n, a positive integer, outputs the factorial representation of n (see Exercise 2.12 and the previous exercise).

10. In checkers, how many different paths (of legal moves) are there from the rightmost square in your back row, to your opponent's back row? Recall that checkers is played on an 8×8 board; your pieces begin only on the black squares and your pieces can only move forward on a diagonal.

11. (*a*) Recall that each j-subset of $\{1, 2, \ldots, n\}$ corresponds with a binary sequence with n bits, exactly j of which are 1s. Given two subsets, say S and T, with S listed before T in lexicographic order, which of the binary sequences is larger (when considered as numbers)?

    (*b*) Design an algorithm that will produce all the j-subsets of an n-set in lexicographic order by manipulating the bit vectors.

12. Let $I_n = \{1, \ldots, n\}$ and j be fixed. Define

$$U = \{B\text{: } B \text{ is a } j\text{-subset of } I_n \text{ that contains } 1\}$$

and

$$V = \{B\text{: } B \text{ is a } j\text{-subset of } I_n \text{ that does not contain } 1\}.$$

Which of U and V is the larger set?

13. Let $I_6 = \{1, 2, 3, 4, 5, 6\}$. How many permutations of I_6 have the even numbers in their correct positions (i.e., 2 is in the second position, etc.)? How many permutations of I_6 have the even numbers in even-numbered positions?

14. How many numbers from 1000 to 3000 have their digits all in the set $\{1, 2, 3, 4, 5\}$? How many such numbers have no repetitions in their digits?

15. Show that Algorithm 3 from Section 5 will always work in at most six guesses. (*Hint:* One way to verify this is using a computer search.)

16. Find an algorithm for Mastermind that uses at most five questions in all cases.

(*Warning:* The solution to this problem is hard enough so that it has been written up in a journal.)*

17. Guess a formula for $\binom{n}{0}^2 + \binom{n}{1}^2 + \cdots + \binom{n}{n}^2$. Prove your guess.

18. How many multiplications are needed to calculate $n!$? How many multiplications and divisions are needed to calculate $\binom{n}{j}$?

19. For $j \leq \lfloor n/2 \rfloor$, find a way to calculate $\binom{n}{j}$ using at most $2j - 2$ multiplications and divisions.

20. Prove the following identity: $\binom{n}{i} = (n/i)\binom{n-1}{i-1}$. Here is another way to calculate $\binom{n}{i}$:

$$\binom{n-i+1}{1} = n - i + 1, \qquad \text{using no multiplications or divisions}$$

$$\binom{n-i+2}{2} = \frac{n-i+2}{2}\binom{n-i+1}{1} \qquad \text{by the identity above}$$

$$\binom{n-i+3}{3} = \frac{n-i+3}{3}\binom{n-i+2}{2} \qquad \text{by the same identity}$$

$$\cdots$$

$$\binom{n-i+i}{i} = \binom{n}{i} = (n/i)\binom{n-1}{i-1} \qquad \text{by repeated application of the identity}$$

Using this approach, determine the number of multiplications and divisions used to calculate $\binom{n}{i}$. Compare this result with your answers in the previous two exercises.

21. Use the results of Exercise 4 to find the expansion of $(1 + x + y)^n$. That is, write this as a polynomial of two variables:

$$a_{0,0} + a_{1,0}x + a_{0,1}y + a_{1,1}xy + a_{2,0}x^2 + a_{0,2}y^2 + \cdots + a_{j,k}x^j y^k + \cdots$$

and find a formula for each coefficient $a_{j,k}$, where $j + k \leq n$.

* Donald Knuth, "The Computer as Master Mind", *Journal of Recreational Mathematics*, Vol. 9(1), 1976–77, pp. 1–6.

22. If $(1 + x + x^2)^n$ is expanded as

$$b_0 + b_1 x + \cdots + b_i x^i + \cdots + b_{2n} x^{2n},$$

then find an expression for each coefficient b_i.

23. Reread the definition of even and odd permutations in Exercises 4.11 and 4.12. Prove that for a permutation p, $d(p, p^*)$, where p^* is the identity permutation, is even if and only if every number of transpositions that transform p into p^* is even.

24. If $p = \langle s_1\ s_2\ \cdots\ s_n \rangle$ is a permutation, we define $Inv(p)$, called the number of **inversions** of p, to be the number of pairs (s_i, s_j) such that $i < j$ and $s_i > s_j$ for $1 \leq i < j \leq n$. Determine $Inv(p)$ for each of the following:

(a) $\langle 1\ 2\ 3\ 4\ 5\ 6 \rangle$
(b) $\langle 2\ 1\ 3\ 4\ 5\ 6 \rangle$
(c) $\langle 2\ 1\ 4\ 3\ 6\ 5 \rangle$
(d) $\langle 6\ 5\ 4\ 3\ 2\ 1 \rangle$

25. Prove that $Inv(p)$ is an even number if and only if p is an even permutation.