THE CLASSIC WORK
NEWLY UPDATED AND REVISED

The Art of Computer Programming

VOLUME 1

Fundamental Algorithms

Third Edition

DONALD E. KNUTH

This page intentionally left blank

Stirling's attempt to generalize the factorial function — and we found a simple derivation of its coefficients in Section 1.2.6 (Example 5). We conclude that

$$\frac{P_{n0}}{n!} = 1 - \frac{1}{1!} + \frac{1}{2!} - \cdots + (-1)^n \frac{1}{n!}. \tag{25}$$

Now let p_{nk} be the probability that a permutation of n objects has exactly k singleton cycles. Since $p_{nk} = P_{nk}/n!$, we have from Eqs. (22) and (25)

$$p_{nk} = \frac{1}{k!}\left(1 - \frac{1}{1!} + \frac{1}{2!} - \cdots + (-1)^{n-k}\frac{1}{(n-k)!}\right). \tag{26}$$

The generating function $G_n(z) = p_{n0} + p_{n1}z + p_{n2}z^2 + \cdots$ is therefore

$$G_n(z) = 1 + \frac{1}{1!}(z-1) + \cdots + \frac{1}{n!}(z-1)^n = \sum_{0\le j\le n} \frac{1}{j!}(z-1)^j. \tag{27}$$

From this formula it follows that $G_n'(z) = G_{n-1}(z)$, and with the methods of Section 1.2.10 we obtain the following statistics on the number of singleton cycles:

$$(\min 0, \quad \text{ave } 1, \quad \max n, \quad \text{dev } 1), \qquad \text{if} \quad n \ge 2. \tag{28}$$

A somewhat more direct way to count the number of permutations having no singleton cycles follows from the *principle of inclusion and exclusion*, which is an important method for many enumeration problems. The general principle of inclusion and exclusion may be formulated as follows: We are given N elements, and M subsets, $S_1, S_2, \ldots, S_M$, of these elements; and our goal is to count how many of the elements lie in none of the subsets. Let $|S|$ denote the number of elements in a set S; then the desired number of objects in none of the sets S_j is

$$N - \sum_{1\le j\le M} |S_j| + \sum_{1\le j<k\le M} |S_j \cap S_k| - \sum_{1\le i<j<k\le M} |S_i \cap S_j \cap S_k| + \cdots + (-1)^M |S_1 \cap \cdots \cap S_M|. \tag{29}$$

(Thus we first subtract the number of elements in $S_1, \ldots, S_M$ from the total number, N; but this underestimates the desired total. So we add back the number of elements that are common to pairs of sets, $S_j \cap S_k$, for each pair S_j and S_k; this, however, gives an overestimate. So we subtract the elements common to triples of sets, etc.) There are several ways to prove this formula, and the reader is invited to discover one of them. (See exercise 25.)

To count the number of permutations on n elements having no singleton cycles, we consider the $N = n!$ permutations and let S_j be the set of permutations in which element j forms a singleton cycle. If $1 \le j_1 < j_2 < \cdots < j_k \le n$, the number of elements in $S_{j_1} \cap S_{j_2} \cap \cdots \cap S_{j_k}$ is the number of permutations in which $j_1, \ldots, j_k$ are singleton cycles, and this is clearly $(n-k)!$. Thus formula (29) becomes

$$n! - \binom{n}{1}(n-1)! + \binom{n}{2}(n-2)! - \binom{n}{3}(n-3)! + \cdots + (-1)^n \binom{n}{n} 0!,$$

in agreement with (25).

The principle of inclusion and exclusion is due to A. de Moivre [see his *Doctrine of Chances* (London: 1718), 61–63; 3rd ed. (1756, reprinted by Chelsea, 1967), 110–112], but its significance was not generally appreciated until it was popularized and developed further by I. Todhunter in his *Algebra* (second edition, 1860), §762, and by W. A. Whitworth in the well-known book *Choice and Chance* (Cambridge: 1867).

Combinatorial properties of permutations are explored further in Section 5.1.

EXERCISES

1. [*02*] Consider the transformation of $\{0,1,2,3,4,5,6\}$ that replaces x by $2x \bmod 7$. Show that this transformation is a permutation, and write it in cycle form.

2. [*10*] The text shows how we might set $(a,b,c,d,e,f) \leftarrow (c,d,f,b,e,a)$ by using a series of replacement operations $(x \leftarrow y)$ and one auxiliary variable t. Show how to do the job by using a series of *exchange* operations $(x \leftrightarrow y)$ and no auxiliary variables.

3. [*03*] Compute the product $\begin{pmatrix} a & b & c & d & e & f \\ b & d & c & a & f & e \end{pmatrix} \times \begin{pmatrix} a & b & c & d & e & f \\ c & d & f & b & e & a \end{pmatrix}$, and express the answer in two-line notation. (Compare with Eq. (4).)

4. [*10*] Express $(a\,b\,d)(e\,f)(a\,c\,f)(b\,d)$ as a product of disjoint cycles.

▶ **5.** [*M10*] Equation (3) shows several equivalent ways to express the same permutation in cycle form. How many different ways of writing that permutation are possible, if all singleton cycles are suppressed?

6. [*M28*] What changes are made to the timing of Program A if we remove the assumption that all blank words occur at the extreme right?

7. [*10*] If Program A is presented with the input (6), what are the quantities X, Y, M, N, U, and V of (19)? What is the time required by Program A, excluding input-output?

▶ **8.** [*23*] Would it be feasible to modify Algorithm B to go from left to right instead of from right to left through the input?

9. [*10*] Both Programs A and B accept the same input and give the answer in essentially the same form. Is the output *exactly* the same under both programs?

▶ **10.** [*M28*] Examine the timing characteristics of Program B, namely, the quantities A, B, $\ldots$, Z shown there; express the total time in terms of the quantities X, Y, M, N, U, V defined in (19), and of F. Compare the total time for Program B with the total time for Program A on the input (6), as computed in exercise 7.

11. [*15*] Find a simple rule for writing π^- in cycle form, if the permutation π is given in cycle form.

12. [*M27*] (*Transposing a rectangular matrix.*) Suppose an $m \times n$ matrix (a_{ij}), $m \neq n$, is stored in memory in a fashion like that of exercise 1.3.2–10, so that the value of a_{ij} appears in location $L + n(i-1) + (j-1)$, where L is the location of a_{11}. The problem is to find a way to *transpose* this matrix, obtaining an $n \times m$ matrix (b_{ij}), where $b_{ij} = a_{ji}$ is stored in location $L + m(i-1) + (j-1)$. Thus the matrix is to be transposed "on itself." (a) Show that the transposition transformation moves the value that appears in cell $L + x$ to cell $L + (mx \bmod N)$, for all x in the range $0 \le x < N = mn - 1$. (b) Discuss methods for doing this transposition by computer.

▶ **13.** [*M24*] Prove that Algorithm J is valid.

▶ **14.** [*M34*] Find the average value of the quantity A in the timing of Algorithm J.

15. [*M12*] Is there a permutation that represents exactly the same transformation both in the canonical cycle form without parentheses and in the linear form?

16. [*M15*] Start with the permutation 1324 in linear notation; convert it to canonical cycle form and then remove the parentheses; repeat this process until arriving at the original permutation. What permutations occur during this process?

17. [*M24*] (a) The text demonstrates that there are $n!\,H_n$ cycles altogether, among all the permutations on n elements. If these cycles (including singleton cycles) are individually written on $n!\,H_n$ slips of paper, and if one of these slips of paper is chosen at random, what is the average length of the cycle that is thereby picked? (b) If we write the $n!$ permutations on $n!$ slips of paper, and if we choose a number k at random and also choose one of the slips of paper, what is the probability that the cycle containing k on that slip is an m-cycle? What is the average length of the cycle containing k?

▸ **18.** [*M27*] What is p_{nkm}, the probability that a permutation of n objects has exactly k cycles of length m? What is the corresponding generating function $G_{nm}(z)$? What is the average number of m-cycles and what is the standard deviation? (The text considers only the case $m = 1$.)

19. [*HM21*] Show that, in the notation of Eq. (25), the number P_{n0} of derangements is exactly equal to $n!/e$ *rounded to the nearest integer*, for all $n \geq 1$.

20. [*M20*] Given that all singleton cycles are written out explicitly, how many different ways are there to write the cycle notation of a permutation that has α_1 one-cycles, α_2 two-cycles, ...? (See exercise 5.)

21. [*M22*] What is the probability $P(n; \alpha_1, \alpha_2, \dots)$ that a permutation of n objects has exactly α_1 one-cycles, α_2 two-cycles, etc.?

▸ **22.** [*HM34*] (The following approach, due to L. Shepp and S. P. Lloyd, gives a convenient and powerful method for solving problems related to the cycle structure of random permutations.) Instead of regarding the number, n, of objects as fixed, and the permutation variable, let us assume instead that we independently choose the quantities $\alpha_1, \alpha_2, \alpha_3, \dots$ appearing in exercises 20 and 21 according to some probability distribution. Let w be any real number between 0 and 1.

a) Suppose that we choose the random variables α_1, α_2, α_3, ... according to the rule that "the probability that $\alpha_m = k$ is $f(w, m, k)$," for some function $f(w, m, k)$. Determine the value of $f(w, m, k)$ so that the following two conditions hold: (i) $\sum_{k\geq 0} f(w, m, k) = 1$, for $0 < w < 1$ and $m \geq 1$; (ii) the probability that $\alpha_1 + 2\alpha_2 + 3\alpha_3 + \cdots = n$ *and* that $\alpha_1 = k_1$, $\alpha_2 = k_2$, $\alpha_3 = k_3$, ... equals $(1 - w)w^n P(n; k_1, k_2, k_3, \dots)$, where $P(n; k_1, k_2, k_3, \dots)$ is defined in exercise 21.

b) A permutation whose cycle structure is α_1, α_2, α_3, ... clearly permutes exactly $\alpha_1 + 2\alpha_2 + 3\alpha_3 + \cdots$ objects. Show that if the α's are randomly chosen according to the probability distribution in part (a), the probability that $\alpha_1 + 2\alpha_2 + 3\alpha_3 + \cdots = n$ is $(1 - w)w^n$; the probability that $\alpha_1 + 2\alpha_2 + 3\alpha_3 + \cdots$ is *infinite* is zero.

c) Let $\phi(\alpha_1, \alpha_2, \dots)$ be any function of the infinitely many numbers α_1, α_2, ... Show that if the α's are chosen according to the probability distribution in (a), the average value of ϕ is $(1 - w)\sum_{n\geq 0} w^n \phi_n$; here ϕ_n denotes the average value of ϕ taken over all permutations of n objects, where the variable α_j represents the number of j-cycles of a permutation. [For example, if $\phi(\alpha_1, \alpha_2, \dots) = \alpha_1$, the value of ϕ_n is the average number of singleton cycles in a random permutation of n objects; we showed in (28) that $\phi_n = 1$ for all $n \geq 1$.]

d) Use this method to find the average number of cycles of *even* length in a random permutation of n objects.

e) Use this method to solve exercise 18.

23. [*HM42*] (Golomb, Shepp, Lloyd.) If l_n denotes the average length of the *longest* cycle in a permutation of n objects, show that $l_n \approx \lambda n + \frac{1}{2}\lambda$, where $\lambda \approx 0.62433$ is a constant. Prove in fact that $\lim_{n\to\infty}(l_n - \lambda n - \frac{1}{2}\lambda) = 0$.

24. [*M41*] Find the variance of the quantity A that enters into the timing of Algorithm J. (See exercise 14.)

25. [*M22*] Prove Eq. (29).

▶ **26.** [*M24*] Extend the principle of inclusion and exclusion to obtain a formula for the number of elements that are in exactly r of the subsets $S_1, S_2, \ldots, S_M$. (The text considers only the case $r = 0$.)

27. [*M20*] Use the principle of inclusion and exclusion to count the number of integers n in the range $0 \le n < am_1m_2\ldots m_t$ that are not divisible by any of $m_1, m_2, \ldots, m_t$. Here m_1, m_2, …, m_t, and a are positive integers, with $m_j \perp m_k$ when $j \ne k$.

28. [*M21*] (I. Kaplansky.) If the "Josephus permutation" defined in exercise 1.3.2–22 is expressed in cycle form, we obtain (1 5 3 6 8 2 4)(7) when $n = 8$ and $m = 4$. Show that this permutation in the general case is the product $(n\ n{-}1\ \ldots\ 2\ 1)^{m-1} \times (n\ n{-}1\ \ldots\ 2)^{m-1} \ldots (n\ n{-}1)^{m-1}$.

29. [*M25*] Prove that the cycle form of the Josephus permutation when $m = 2$ can be obtained by first expressing the "perfect shuffle" permutation of $\{1, 2, \ldots, 2n\}$, which takes $(1, 2, \ldots, 2n)$ into $(2, 4, \ldots, 2n, 1, 3, \ldots, 2n{-}1)$, in cycle form, then reversing left and right and erasing all the numbers greater than n. For example, when $n = 11$ the perfect shuffle is (1 2 4 8 16 9 18 13 3 6 12)(5 10 20 17 11 22 21 19 15 7 14) and the Josephus permutation is (7 11 10 5)(6 3 9 8 4 2 1).

30. [*M24*] Use exercise 29 to show that the fixed elements of the Josephus permutation when $m = 2$ are precisely the numbers $(2^d - 1)(2n + 1)/(2^{d+1} - 1)$ for all positive integers d such that this is an integer.

31. [*HM38*] Generalizing exercises 29 and 30, prove that the jth man to be executed, for general m and n, is in position x, where x may be computed as follows: Set $x \leftarrow jm$; then, while $x > n$, set $x \leftarrow \lfloor (m(x - n) - 1)/(m - 1) \rfloor$. Consequently the average number of fixed elements, for $1 \le n \le N$ and fixed $m > 1$ as $N \to \infty$, approaches $\sum_{k\ge1}(m - 1)^k/(m^{k+1} - (m - 1)^k)$. [Since this value lies between $(m - 1)/m$ and 1, the Josephus permutations have slightly fewer fixed elements than random ones do.]

32. [*M25*] (a) Prove that any permutation $\pi = \pi_1\pi_2\ldots\pi_{2m+1}$ of the form

$$\pi = (2\ 3)^{e_2}(4\ 5)^{e_4}\ldots(2m\ 2m{+}1)^{e_{2m}}(1\ 2)^{e_1}(3\ 4)^{e_3}\ldots(2m{-}1\ 2m)^{e_{2m-1}},$$

where each e_k is 0 or 1, has $|\pi_k - k| \le 2$ for $1 \le k \le 2m + 1$.

(b) Given any permutation ρ of $\{1, 2, \ldots, n\}$, construct a permutation π of the stated form such that $\rho\pi$ is a single cycle. Thus every permutation is "near" a cycle.

33. [*M33*] If $m = 2^{2^l}$ and $n = 2^{2l+1}$, show how to construct sequences of permutations $(\alpha_{j1}, \alpha_{j2}, \ldots, \alpha_{jn}; \beta_{j1}, \beta_{j2}, \ldots, \beta_{jn})$ for $0 \le j < m$ with the following "orthogonality" property:

$$\alpha_{i1}\beta_{j1}\alpha_{i2}\beta_{j2}\ldots\alpha_{in}\beta_{jn} = \begin{cases} (1\,2\,3\,4\,5), & \text{if } i = j; \\ (), & \text{if } i \ne j. \end{cases}$$

Each α_{jk} and β_{jk} should be a permutation of $\{1, 2, 3, 4, 5\}$.

▶ **34.** [*M25*] (*Transposing blocks of data.*) One of the most common permutations needed in practice is the change from $\alpha\beta$ to $\beta\alpha$, where α and β are substrings of an array.

In other words, if $x_0x_1 \ldots x_{m-1} = \alpha$ and $x_m x_{m+1} \ldots x_{m+n-1} = \beta$, we want to change the array $x_0x_1 \ldots x_{m+n-1} = \alpha\beta$ to the array $x_m x_{m+1} \ldots x_{m+n-1} x_0 x_1 \ldots x_{m-1} = \beta\alpha$; each element x_k should be replaced by $x_{p(k)}$ for $0 \le k < m+n$, where $p(k) = (k+m) \bmod (m+n)$. Show that every such "cyclic-shift" permutation has a simple cycle structure, and exploit that structure to devise a simple algorithm for the desired rearrangement.

35. [*M30*] Continuing the previous exercise, let $x_0x_1 \ldots x_{l+m+n-1} = \alpha\beta\gamma$ where α, β, and γ are strings of respective lengths l, m, and n, and suppose that we want to change $\alpha\beta\gamma$ to $\gamma\beta\alpha$. Show that the corresponding permutation has a convenient cycle structure that leads to an efficient algorithm. [Exercise 34 considered the special case $m = 0$.] *Hint:* Consider changing $(\alpha\beta)(\gamma\beta)$ to $(\gamma\beta)(\alpha\beta)$.

36. [*27*] Write a `MIX` subroutine for the algorithm in the answer to exercise 35, and analyze its running time. Compare it with the simpler method that goes from $\alpha\beta\gamma$ to $(\alpha\beta\gamma)^R = \gamma^R\beta^R\alpha^R$ to $\gamma\beta\alpha$, where σ^R denotes the left-right reversal of the string σ.

37. [*M26*] (*Even permutations.*) Let π be a permutation of $\{1, \ldots, n\}$. Prove that π can be written as the product of an even number of 2-cycles if and only if π can be written as the product of exactly two n-cycles.

▶ **38.** [*M22*] (*Conjugate permutations.*) When π and τ are arbitrary permutations, the permutation

$$\pi^\tau = \tau^- \pi \tau$$

is called "the conjugate of π by τ."

a) Prove that $\pi = \pi^\tau$ if and only if π commutes with τ.
b) Suppose π has the cycle form $(a_{11} \ldots a_{1l_1}) \ldots (a_{k1} \ldots a_{kl_k})$. Prove that π^τ has the cycle form $(b_{11} \ldots b_{1l_1}) \ldots (b_{k1} \ldots b_{kl_k})$, where τ takes $a_{ij} \mapsto b_{ij}$.
c) Find all τ for which $\bigl((1\,2\,3)(4\,5)(6\,7\,8)\bigr)^\tau = (8\,7\,6)(5\,4)(3\,2\,1)$.
d) Find all τ for which $\bigl((1\,2\,3)(4\,5)(6\,7\,8)\bigr)^\tau = (8\,7\,6\,5)(4\,3\,2\,1)$.
e) True or false: Every permutation π is conjugate to its inverse, π^-.
f) True or false: $(\pi\sigma)^\tau = \pi^\tau\sigma^\tau$ and $\pi^{(\sigma\tau)} = (\pi^\sigma)^\tau$, for all π, σ, and τ.

1.4. SOME FUNDAMENTAL PROGRAMMING TECHNIQUES

1.4.1. Subroutines

When a certain task is to be performed at several different places in a program, it is usually undesirable to repeat the coding in each place. To avoid this situation, the coding (called a *subroutine*) can be put into one place only, and a few extra instructions can be added to restart the outer program properly after the subroutine is finished. Transfer of control between subroutines and main programs is called *subroutine linkage.*

Each machine has its own peculiar manner for achieving efficient subroutine linkage, usually involving special instructions. In MIX, the J-register is used for this purpose; our discussion will be based on MIX machine language, but similar remarks will apply to subroutine linkage on other computers.

Subroutines are used to save space in a program; they do not save any time, other than the time implicitly saved by occupying less space — for example, less time to load the program, or fewer passes necessary in the program, or better use of high-speed memory on machines with several grades of memory. The extra time taken to enter and leave a subroutine is usually negligible.

Subroutines have several other advantages. They make it easier to visualize the structure of a large and complex program; they form a logical segmentation of the entire problem, and this usually makes debugging of the program easier. Many subroutines have additional value because they can be used by people other than the programmer of the subroutine.

Most computer installations have built up a large library of useful subroutines, and such a library greatly facilitates the programming of standard computer applications that arise. A programmer should not think of this as the *only* purpose of subroutines, however; subroutines should not always be regarded as general-purpose programs to be used by the community. Special-purpose subroutines are just as important, even when they are intended to appear in only one program. Section 1.4.3.1 contains several typical examples.

The simplest subroutines are those that have only one entrance and one exit, such as the MAXIMUM subroutine we have already considered (see Section 1.3.2, Program M). For reference, we will recopy that program here, changing it so that a fixed number of cells, 100, is searched for the maximum:

```
* MAXIMUM OF X[1..100]
MAX100  STJ   EXIT   Subroutine linkage
        ENT3  100    M1. Initialize.
        JMP   2F
1H      CMPA  X,3    M3. Compare.
        JGE   *+3                                      (1)
2H      ENT2  0,3    M4. Change m.
        LDA   X,3    New maximum found
        DEC3  1      M5. Decrease k.
        J3P   1B     M2. All tested?
EXIT    JMP   *      Return to main program. ▮
```