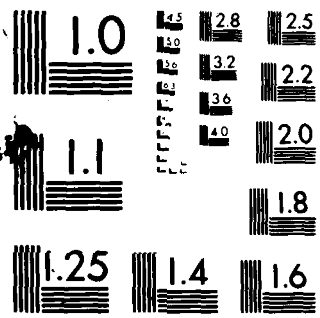
AD-A112 929 STANFORD UNIV CA DEPT OF COMPUTER SCIENCE F/G 9/2
THE LAST WHOLE ERRATA CATALOG,(U)
JUL 81 D E KNUTH N00014-81-K-0269
UNCLASSIFIED STAN-CS-81-868 NL

END
DATE FILMED
4 82
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

July 1981

Report. No. STAN-CS-81-868

The Last Whole Errata Catalog

by

Donald E. Knuth

Research sponsored by

Office of Naval Research
and
National Science Foundation

Department of Computer Science

Stanford University
Stanford, CA 94305

THE LAST WHOLE ERRATA CATALOG

by Donald E. Knuth

Stanford University

The publication of this report was supported in part by National Science Foundation grant MCS-77-23738, by National Science Foundation grant IST-79-21977, and by Office of Naval Research contract N00014-81-K-0269. Reproduction in whole or in part is permitted for any purpose of the United States government.

THE ART OF COMPUTER PROGRAMMING

E • R • R • A • T • A et A • D • D • E • N • D • A

July 13, 1981

This list supplements previous errata published in Stanford reports CS551 (1976) and CS712 (1979). It includes the first corrections and changes to the second edition of volume two (published January, 1981) as well as to the most recent printings of volumes one and three (first published in 1975). In addition to the errors listed here, about half of the occurrences of 'which' in volumes one and three should be changed to 'that'.

1.ix **line -7** 10/10/79 *1*

historically have always developed from $\rightsquigarrow$
almost always owe their origin to

1.xx **line -5** 1/5/81 *2*

2.2 $\rightsquigarrow$ 2.2.

1.1 ***historical improvements*** 9/4/79 *3*

lines -6, -4: Khowârizmî $\rightsquigarrow$ Khwârizmî
lines -5, -4: Khowârizm." ... Khiva. $\rightsquigarrow$ Khwârizm." The Aral Sea in Central Asia was once known as Lake Khwârizm, and the Khwârizm region is located in the Amu River basin just south of that sea.
line -3: *w'al-muqabala* $\rightsquigarrow$ *wa'l-muqâbala*
line -3: restoration and reduction $\rightsquigarrow$ restoring and equating
lines -2, -1: although ... algebraic. $\rightsquigarrow$ which was a systematic study of the solution of linear and quadratic equations.

1.25 **exercise 19** 1/26/80 *4*

a 14-digit integer, $\rightsquigarrow$ an integer whose decimal representation is 14 digits long,

1.42 **line 4** 2/23/81 *5*

$\sum_{1\le k<n}$ $\rightsquigarrow$ $\sum_{1\le k\le n}$

1.61 **lines 4 and 5** 6/1/81 *6*

to introduce *still further* complication $\rightsquigarrow$ to complicate things even more

1.72 **line -4 (overrides 1979 change #18)** 8/30/80 *7*

$A_{n(k-1)} + \binom{n}{k}$. $\rightsquigarrow$ $A_{(n-1)(k-1)} + \binom{n}{k}$, for $nk > 0$.

1.78 **line -2** 9/4/79 *8*

al-Khowârizmî $\leadsto$ al-Khwârizmî

1.86 **line -12** 12/16/79 *9*

$|z| < z_0$. $\leadsto$ $|z| < |z_0|$.

1.87 **three lines after (4)** 10/26/79 *10*

latter $\leadsto$ last-mentioned

1.88 **bottom line** 4/1/79 *11*

$1 \le j < m$ $\leadsto$ $0 \le j < m$

1.97 **clarifying remarks** 3/10/81 *12*

line 10: $A = k$. $\leadsto$ $A = k$. Let this number be P_{nk}.
line 14: that $\leadsto$
that $P_{nk} = P_{(n-1)(k-1)} + (n-1)P_{(n-1)k}$, which leads to

1.108 **line 7** 9/26/80 *13*

Academæ $\leadsto$ *Academiæ*

1.110 **just after (13), overriding 1976 change #31** 10/25/79 *14*

provided that ... *to n.* $\leadsto$ *provided that* $f^{(2k+2)}(x)f^{(2k+4)}(x) > 0$ for $1 < x < n$.

1.112 **new wording for exercise 3** 10/25/79 *15*

3. [*HM20*] Let $C_m = ((-1)^m B_m/m!)(f^{(m-1)}(n) - f^{(m-1)}(1))$ be the mth correction term in Euler's summation formula. If $f^{(2k)}(x)$ has a constant sign for $1 \le x \le n$, show that $|R_{2k}| \le |C_{2k}|$ when $k > 0$; in other words, the remainder is not larger in absolute value than the *last* term computed.

1.119 **new exercise** 3/16/81 *16*

18. [*M25*] Show that the sums $\sum \binom{n}{k} k^k (n-k)^{n-k}$ and $\sum \binom{n}{k}(k+1)^k (n-k)^{n-k}$ can be expressed very simply in terms of the Q function.

1.122 **improvements in wording** 6/4/80 *17*

line 1: *A* ... position has $\leadsto$ *A computer word consists of five bytes and a sign*. The sign portion has
line 8: bytes, and its sign $\leadsto$ bytes; it behaves as if its sign
line 17: the preceding "JUMP" instruction, $\leadsto$ the most recent "jump" operation,

1.123 **more improvements in wording** 4/12/81 *18*

line 2 after (3): 8 is $\leadsto$ 8 specifies
lines 10 and 11 after (3): address of an instruction. $\leadsto$ effective address.
lines 13 and 14 after (3): address of the instruction. $\leadsto$ address.

1.132 **wrong fonts** 6/4/80 *19*

line -17: A through Z ↝ A through Z
line -16: 0, 1, ..., 9; ↝ 0, 1, ..., 9;
line -12: Φ and Π ↝ Δ, Σ, and Π

1.132 **line -9** 3/30/81 *20*

ignored. ↝ ignored. When a typewriter is used for input, the "carriage return" that is typed at the end of each line causes the remainder of that line to be filled with blanks.

1.136 **and also page 137** 6/6/80 *21*

replace by the chart on the endpapers of the new volume 2

1.140 **line -3** 6/6/80 *22*

bytes 20, ... since ↝ bytes 10, 20, 21, 49, 50, ... (i.e., the characters Δ, Σ, Π, $, <, ...) since

1.141 **line 13** 6/4/80 *23*

cell(X + i). ↝ CONTENTS(X + i).

1.148 **changes brought about by the demise of punched cards** 3/30/81 *24*

Fig. 15 will change to include also the following copy as typed on a typical hardcopy terminal:

```
* EXAMPLE PROGRAM ... TABLE OF PRIMES
*
L  EQU 500
PRINTER EQU 18
```

The caption will change to "... onto cards, or typed on a terminal."
line -6: cards, ↝ cards or typed on a computer terminal,
line -5: used: ↝ used in the case of punched cards:

1.149 **new paragraph to follow line 5** 3/30/81 *25*

When the input comes from a terminal, a less restrictive format is used: The LOC field ends with the first blank space, while the OP and ADDRESS fields (if present) begin with a nonblank character and continue to the next blank; the special OP code ALF, however, is followed by either two blank spaces and five characters of alphameric data, or by a single blank space and five alphameric characters, the first of which is nonblank. The remainder of each line contains optional remarks.

1.150 **line 22** 6/1/81 *26*

context), ↝ OP field, as shown in Table 1.3.1–1),

1.151 **lines 9 and 10** 6/4/80 *27*

values: C, F, A, and I; the ↝ values: C, F, A, and I. The

1.173 **new material for this page and the following one** 11/11/80 *28*

here is a new Algorithm I together with a new Program I:

Algorithm I (*Inverse in place*). Replace $X[1]X[2]\ldots X[n]$, a permutation on $\{1,2,\ldots,n\}$, by its inverse. This algorithm is due to Huang Bing-Chao.

I1. [Initialize.] Set $m \leftarrow n$, $j \leftarrow -1$.

I2. [Next element.] Set $i \leftarrow X[m]$. If $i < 0$, go to step I5 (the element has already been processed).

I3. [Invert one.] (At this point $j < 0$ and $i = X[m]$. If m is not the largest element of its cycle, the original permutation had $X[-j] = m$.) Set $X[m] \leftarrow j$, $j \leftarrow -m$, $m \leftarrow i$, $i \leftarrow X[m]$.

I4. [End of cycle?] If $i > 0$, go back to I3 (the cycle has not ended); otherwise set $i \leftarrow j$. (In the latter case, the original permutation has $X[-j] = m$, and m is largest in its cycle.)

I5. [Store final value.] Set $X[m] \leftarrow -i$. (Originally $X[i]$ was equal to m.)

I6. [Loop on m.] Decrease m by 1. If $m > 0$, go back to I2; otherwise the algorithm terminates. ▮

For an example of this algorithm, see Table 2. The method is based on inversion of successive cycles of the permutation, tagging the inverted elements by making them negative, afterwards restoring the correct sign.

Table 2

COMPUTING THE INVERSE OF 621543 BY ALGORITHM I

(Read columns from left to right.) At point *, the cycle (163) has been inverted.

After step:	I2	I3	I3	I3	I5*	I2	I3	I3	I5	I2	I5	I5	I3	I5	I5
$X[1]$	6	6	6	−3	−3	−3	−3	−3	−3	−3	−3	−3	−3	−3	3
$X[2]$	2	2	2	2	2	2	2	2	2	2	2	2	−4	2	2
$X[3]$	1	1	−6	−6	−6	−6	−6	−6	−6	−6	−6	6	6	6	6
$X[4]$	5	5	5	5	5	5	5	−5	−5	−5	5	5	5	5	5
$X[5]$	4	4	4	4	4	4	−1	−1	4	4	4	4	4	4	4
$X[6]$	3	−1	−1	−1	1	1	1	1	1	1	1	1	1	1	1
m	6	3	1	6	6	5	4	5	5	4	4	3	2	2	1
j	−1	−6	−3	−1	−1	−1	−5	−4	−4	−4	−4	−4	−2	−2	−2
i	3	1	6	−1	−1	4	5	−1	−4	−5	−5	−6	−4	−2	−3

Algorithm I resembles parts of Algorithm A, and it very strongly resembles the cycle-finding algorithm in Program B (lines 50–64). Thus it is typical of a number of algorithms involving rearrangements. When preparing a MIX implementation, we find that it is most convenient to keep the value of $-i$ in a register instead of i itself:

Program I (*Inverse in place*). $rI1 \equiv m$; $rI2 \equiv -i$; $rI3 \equiv j$; and $n = $ N, a symbol to be defined when this program is assembled as part of a larger routine.

```
INVERT ENT1 N     1  I1. Initialize. m ← n.
       ENT3 -1    1  j ← −1.
2H     LD2N X,1   N  I2. Next element. i ← X[m].
       J2P  5F    N  To I6 if i < 0.
3H     ST3  X,1   N  I3. Invert one. X[m] ← j.
       ENN3 0,1   N  j ← −m.
       ENN1 0,2   N  m ← i.
       LD2N X,1   N  i ← X[m].
4H     J2N  3B    N  End of cycle? To I3 if i > 0.
       ENN2 0,3   C  Otherwise set i ← j.
5H     ST2  X,1   N  I5. Store final value. X[m] ← −i.
6H     DEC1 1     N  I6. Loop on m.
       J1P  2B    N  To I2 if m > 0. ▮
```

The timing for this program is easily worked out in the manner shown earlier; every element $X[m]$ is set first to a negative value in step I3 and later to a positive value in step I5. The total time comes to $(14N + C + 2)u$, where N is the order of the permutation and C is the total number of cycles. The behavior of C in a random permutation is analyzed below.

There is almost always more than one algorithm ...

1.177 **line 17** 11/11/80 *29*

A, B, and I, $\leadsto$ A and B,

1.209 **program line 21** 4/4/80 *30*

LDA $\leadsto$ ENTA

1.234 **line -17** 3/3/81 *31*

i.e., $\leadsto$ e.g.,

1.246 **improved overlap** 2/4/79 *32*

line -10 should become: OLDTOP$[j] \equiv$ D$[j] \equiv$ NEWBASE$[j+1]$
line -9: $n+1$; $\leadsto$ n;
lines -8 and -7: delete the sentence "It will ... overlap."

1.248 **addendum to 1979 change #47** 2/7/79 *33*

See also A. S. Fraenkel, *Inf. Proc. Letters* 8 (1979), 9–10, who suggests working with pairs of stacks that grow towards each other.

1.250 **new rating for exercise 13** 3/1/79 *34*

[*M47*] $\leadsto$ [*HM44*]

1.252 **lines -12 and -11** 8/18/80 *35*

together or to break one apart. $\leadsto$ together, or to break one apart into two that will grow independently.

1.254 **replacement for lines 16 and 17** 2/4/79 *36*

Otherwise set X $\leftarrow$ POOLMAX and POOLMAX $\leftarrow$ POOLMAX + c, (7)
where c is the node size;
OVERFLOW now occurs if POOLMAX > SEQMIN."

1.284 **the line for time 0693** 7/1/79 *37*

M1 $\leadsto$ M5

1.309 **line 10** 9/4/80 *38*

and two $\leadsto$ *and the elements of two*

1.323 **trivial improvements to Program S** 10/17/79 39

line 03: ENT6 $\rightsquigarrow$ ENT5
line 03: Q $\rightsquigarrow$ P
line 04: S2 $\rightsquigarrow$ 2F
line 09: $n+1$ $\rightsquigarrow$ n
line 09: Set $\rightsquigarrow$ <u>S2. Search to left.</u> Set
line 10, first column: $\rightsquigarrow$ 2H
line 11: *-2 $\rightsquigarrow$ S2

1.324 **line 5** 10/17/79 40

8 $\rightsquigarrow$ 7

1.381 **new exercise** 5/19/81 41

27. [*M30*] (*Steady states.*) Let G be a directed graph on vertices $V_1, \ldots, V_n$, whose arcs have been assigned probabilities $p(e)$ as in exercise 26. Instead of having "start" and "stop" vertices, however, assume that G is strongly connected; thus, each vertex V_j is a root, and we assume that the probabilities $p(e)$ are positive and satisfy $\sum_{\mathrm{init}(e)=V_j} p(e) = 1$ for all j. A random process of the kind described in exercise 26 is said to have a "steady state" $(x_1, \ldots, x_n)$ if

$$x_j = \sum_{\mathrm{fin}(e)=V_j} p(e)x_{\mathrm{init}(e)}, \qquad 1 \le j \le n.$$

Let t_j be the sum, over all oriented subtrees T_j of G that are rooted at V_j, of the products $\prod_{e\in T_j} p(e)$. Prove that $(t_1, \ldots, t_n)$ is a steady state of the random process.

1.402 **three lines before (9)** 3/19/81 42

Huffman: $\rightsquigarrow$ Huffman [*Proc. IRE* **40** (1951), 1098–1101]:

1.404 **lines 1 through 5** 3/15/81 43

In general, ... method has $\rightsquigarrow$

Every time this construction combines two weights, they are at least as big as the weights previously combined, if the given w_i were nonnegative. This means that there is a neat way to find Huffman's tree, provided that the given weights have been sorted into nondecreasing order: We simply maintain two queues, one containing the original weights and the other containing the combined weights. At each step the smallest unused weight will appear at the front of one of the queues, so we never have to search for it. See exercise 13, which shows that the same idea works even when the weights may be negative.

In general, there are many trees that minimize $\sum w_j l_j$. If the algorithm sketched in the preceding paragraph always uses an original weight instead of a combined weight in case of ties, then the tree it constructs has

1.405 **second line of exercise 10** 3/15/81 44

given weights $\rightsquigarrow$ given nonnegative weights

1.405 **rating for exercise 12 (overrides 1976 change #81)** 3/15/81 45

Suppose $\rightsquigarrow$ [*M20*] Suppose

1.405 new exercises 3/15/81 46

13. [*22*] Design an algorithm that begins with m weights $w_1 \le w_2 \le \cdots \le w_m$ and constructs an extended binary tree having minimum weighted path length. Represent the final tree in three arrays

$$A[1], \ldots, A[2m-1]; \quad L[1], \ldots, L[m-1]; \quad R[1], \ldots, R[m-1];$$

here $L[i]$ and $R[i]$ point to the left and right sons of internal node i, the root is node 1, and $A[i]$ is the weight of node i. The original weights should appear as the external node weights $A[m], \ldots, A[2m-1]$. Your algorithm should make fewer than $2m$ weight-comparisons. *Caution:* Some or all of the given weights may be negative!

14. [*25*] (T. C. Hu and A. C. Tucker.) After k steps of Huffman's algorithm, the nodes combined so far form a forest of $m-k$ extended binary trees. Prove that this forest has the smallest total weighted path length, among all forests of $m-k$ extended binary trees that have the given weights.

15. [*M25*] Show that a Huffman-like algorithm will find an extended binary tree that minimizes (a) $\max(w_1 + l_1, \ldots, w_m + l_m)$; (b) $w_1 x^{l_1} + \cdots + w_m x^{l_m}$, given $x > 1$.

16. [*M25*] (F. K. Hwang.) Let $w_1 \le \cdots \le w_m$ and $w'_1 \le \cdots \le w'_m$ be two sets of weights with

$$\sum_{1 \le j \le k} w_j \le \sum_{1 \le j \le k} w'_j \qquad \text{for } 1 \le k \le m.$$

Prove that the minimum weighted path lengths satisfy $\sum_{1 \le j \le m} w_j l_j \le \sum_{1 \le j \le m} w'_j l'_j$.

17. [*HM30*] (C. R. Glassey and R. M. Karp.) Let $s_1, \ldots, s_{m-1}$ be the numbers inside the internal (circular) nodes of an extended binary tree formed by Huffman's algorithm, in the order of construction. Let $s'_1, \ldots, s'_{m-1}$ be the internal node weights of any extended binary tree on the same set of weights $\{w_1, \ldots, w_m\}$, listed in any order such that each non-root internal node appears before its father. (a) Prove that $\sum_{1 \le j \le k} s_j \le \sum_{1 \le j \le k} s'_j$ for $1 \le k < m$. (b) The result of (a) is equivalent to

$$\sum_{1 \le j < m} f(s_j) \le \sum_{1 \le j < m} f(s'_j)$$

for every nondecreasing concave function f, i.e., every function f with $f'(x) \ge 0$ and $f''(x) \le 0$. [Cf. Hardy, Littlewood, and Polya, *Messenger of Math.* 58 (1929), 145–152.] Use this fact to study the recurrence

$$F(n) = f(n) + \min_{1 \le k < n} \bigl(F(k) + F(n-k)\bigr), \qquad F(1) = 0,$$

given any function $f(n)$ such that $\Delta f(n) = f(n+1) - f(n) \ge 0$ and $\Delta^2 f(n) = \Delta f(n+1) - \Delta f(n) \le 0$.

1.420 new paragraph before the exercises 2/7/79 47

Daniel P. Friedman and David S. Wise have observed that the reference counter method can be employed satisfactorily in many cases even when lists point to themselves, if certain link fields are not included in the counts [*Inf. Proc. Letters* 8 (1979), 41–45].

1.448 line 6 after the caption 4/6/81 48

changed from  changed to vary from

1.449 **lines -7 through -4** 5/21/81 49

algorithms ... and here are $\leadsto$ methods that are recommended as a consequence of the remarks above: (i) the boundary tag system, as modified in exercises 12 and 16; and (ii) the buddy system. Here are

1.451 **bottom line** 3/20/81 50

36–40. $\leadsto$ 36–40, and in exercises 42–43 where he has shown that the best-fit method has a very bad worst case by comparison with first-fit.

1.455 **new exercises for bottom of page** 4/1/81 51

42. [*M40*] (J. M. Robson, 1975.) Let $N_{\rm BF}(n,m)$ be the amount of memory needed to guarantee non-overflow when the best-fit method is used for allocation (cf. exercise 38). Find an attacking strategy to show that $N_{\rm BF}(n,m) \geq nm - O(n+m^2)$.

43. [*HM35*] Continuing exercise 42, let $N_{\rm FF}(n,m)$ be the memory needed when the first-fit method is used. Show that $N_{\rm FF}(n,m) \leq nH_m/\ln 2$, so the worst case of first-fit is not far from the best possible worst case.

1.463 **correction to 1979 change #73** 2/14/79 52

Such graph machines ... fixed. $\leadsto$ Linking automata can easily simulate graph machines, taking at most a bounded number of steps per graph step. Conversely, however, it is unlikely that graph machines can simulate arbitrary linking automata without unboundedly increasing the running time, unless the definition is changed from undirected to directed graphs, in view of the restriction to vertices of bounded degree.

1.472 **first two lines** 7/8/81 53

Note: The formulas ... differences." $\leadsto$ *Notes:* Dr. Matrix was anticipated in this discovery by L. Euler in 1762; see Euler's *Opera Omnia*, ser. 1, vol. 6, 486–493.

1.474 **line 7** 6/25/81 54

$i+n-1$, and $j+n-1$. $\leadsto$ $i+n-1$, $j+n-1$, $n-i+1$, and $n-j+1$.

1.478 **answer 41** 1/5/80 55

line -2: i.e. $\leadsto$ i.e.,
line -1: are ... 2⌋. $\leadsto$
are $\lceil\sqrt{2n}-\frac{1}{2}\rceil$, $\lceil(-1+\sqrt{1+8n})/2\rceil$, $\lfloor(1+\sqrt{8n-7})/2\rfloor$, etc.

1.488 **line 1 of answer 52** 1/10/81 56

$\pi^2/6-1$. $\leadsto$ $\pi^2/6$.

1.488 **line 3 of answer 58** 10/20/79 57

$q^{(s-n-k)k}$ $\leadsto$ $q^{(s-n+k)k}$

1.488 **new answer to exercise 59** 8/30/80 58

59. $(n+1)\binom{n}{k}-\binom{n}{k+1}$.

1.498 new answer to 1.2.11.2-3, overrides 1976 change #104 10/15/79 *59*

3. $|R_{2k}| \le |B_{2k}/(2k)!| \int_1^n |f^{(2k)}(x)\,dx|$. [*Notes:* We have $B_m(x) = (-1)^m B_m(1-x)$, and $B_m(x)$ is $m!$ times the coefficient of z^m in $ze^{xz}/(e^z-1)$. In particular, since $e^{z/2}/(e^z-1) = 1/(e^{z/2}-1) - 1/(e^z-1)$ we have $B_m(\frac{1}{2}) = (2^{1-m}-1)B_m$. It is not difficult to prove that the maximum of $|B_{2m} - B_{2m}(x)|$ for $0 \le x \le 1$ occurs at $x = \frac{1}{2}$. Now when $k \ge 2$ we have $R_{2k-2} = C_{2k} + R_{2k} = \int_1^n (B_{2k} - B_{2k}(\{x\}))f^{(2k)}(x)\,dx/(2k)!$, and $B_{2k} - B_{2k}(\{x\})$ is between 0 and $(2-2^{1-2k})B_{2k}$, hence R_{2k-2} lies between 0 and $(2-2^{1-2k})C_{2k}$. It follows that R_{2k} lies between $-C_{2k}$ and $(1-2^{1-2k})C_{2k}$, a slightly stronger result. According to this argument we see that if $f^{(2k+2)}(x)f^{(2k+4)}(x) > 0$ for $1 < x < n$, the quantities C_{2k+2} and C_{2k+4} have opposite signs, while R_{2k} has the sign of C_{2k+2} and R_{2k+2} has the sign of C_{2k+4} and $|R_{2k+2}| \le |C_{2k+2}|$; this proves (13). Cf. J. F. Steffensen, *Interpolation* (Baltimore: 1927), §14.]

1.499 exercise 7 (overrides 1979 change #80) 3/25/81 *60*

(It is "Glaisher's constant" 1.2824271...) To $\rightsquigarrow$ To

This formula ... $n = 4$. $\rightsquigarrow$

(The constant A is "Glaisher's constant" 1.28242..., which equals $(2\pi e^{\gamma-\zeta'(2)/\zeta(2)})^{1/12}$; cf. F. W. J. Olver, *Asymptotics and Special Functions* (New York: Academic Press, 1974), Section 8.3.3.)

1.501 new answer 3/16/81 *61*

18. Let $S_n(x,y) = \sum \binom{n}{k}(x+k)^k(y+n-k)^{n-k}$. Then for $n > 0$ we have $S_n(x,y) = x\sum\binom{n}{k}(x+k)^{k-1}(y+n-k)^{n-k} + n\sum\binom{n-1}{k}(x+1+k)^k(y+n-1-k)^{n-1-k} = (x+y+n)^n + nS_{n-1}(x+1,y)$ by Abel's formula 1.2.6-16; consequently $S_n(x,y) = \sum\binom{n}{k}k!(x+y+n)^{n-k}$. [This formula is due to Cauchy, who proved it by quite different means in *Exercices de Mathématiques* (Paris: 1826), 62–73.] The stated sums are therefore equal respectively to $n^n(1+Q(n))$ and $(n+1)^nQ(n+1)$.

1.510 answer 13 6/4/80 *62*

line 2, replace by two lines: `TAPE EQU 19` Input unit number
`TYPE EQU 19` Output unit number

lines 16 and 18: `UNIT` $\rightsquigarrow$ `TAPE` (twice)

lines 38 and 42 (the latter is on page 511): `19` $\rightsquigarrow$ `TYPE` (twice)

1.515 line 5 10/18/79 *63*

For ... history, $\rightsquigarrow$

Historical notes: C. Haros gave a (more complicated) rule for constructing such sequences, in *J. de l'École Polytechnique* 4, 11 (1802), 364–368; his method was correct, but his proof was inadequate. The geologist John Farey independently conjectured several years later that x_k/y_k is always equal to $(x_{k-1}+x_{k+1})/(y_{k-1}+y_{k+1})$ [*Philos. Magazine and Journal* 47 (1816), 385–386]; a proof was supplied shortly afterwards by A. Cauchy [*Bull. Société Philomathique de Paris* (3) 3 (1816), 133–135], who attached Farey's name to the series. For more of its interesting properties,

1.531 **line -2** 10/18/79 64

X's. For the history of the ballot problem ⟿ X's. This problem was actually resolved as early as 1708 by Abraham de Moivre, who showed that the number of sequences containing l A's and m B's, and containing at least one initial substring with n more A's than B's, is $f(l,m,n) = \binom{l+m}{\min(m,l-n)}$. In particular, $a_n = \binom{2n}{n} - f(n,n,1)$ as above. (De Moivre stated this result without proof [*Philos. Trans.* **27** (1711), 262–263]; but it is clear from other passages in his paper that he knew how to prove it, since the formula is obviously true when $l \geq m+n$, and since his generating-function approach to similar problems yields the symmetry condition $f(l,m,n) = f(m+n, l-n, n)$ by simple algebra.) For the later history of the ballot problem

1.538 **insert new answer** 3/1/79 65

13. A. C. Yao has shown that $\max(k_1,k_2)$ will be $\frac{1}{2}m + \left(2\pi(1-2p)\right)^{-1/2}\sqrt{m} + O\left(m^{-1/2}(\log m)^2\right)$ for large m, when $p < \frac{1}{2}$. [*SIAM J. Computing* **10** (1981), 398–403.]

1.547 **answer 5** 3/3/81 66

(Solution by B. Young.) ⟿ (Cf. exercise 2.2.3–7.)

1.548 **first line of answer 9** 4/17/79 67

should. ⟿ should; except in the instructive anomalous case that COEF = 0 for some term with ABC $\geq$ 0, when it fails badly.

1.550 **exercise 18 (corrects 1979 change #96)** 3/2/77 68

denotes, ... are included ⟿ denotes "exclusive or." Other invertible operations, such as addition or subtraction modulo the pointer field size, could also be used. It is convenient to include two adjacent list heads

1.560 **additional sentence to follow 1976 change #135** 1/17/79 69

(Steps T4 and T5 can be streamlined so that nodes are not taken off the stack and immediately reinserted.)

1.562 **answer 21** 10/17/79 70

21. The following ⟿ .

21. (Solution by Ď. Branislav, traverses either in preorder or inorder.)

U1. [Initialize.] If $T = \Lambda$, terminate the algorithm. Otherwise set $Q \leftarrow T$.

U2. [Preorder visit.] If traversing in preorder, visit NODE(Q).

U3. [Go to left.] Set R ← LLINK(Q). If R = Λ, go to U5.

U4. [Insert a right thread.] Set P ← Q and Q ← R, then set R ← RLINK(R) zero or more times until RLINK(R) = Λ. Set RTAG(R) ← "−" and RLINK(R) ← P. Return to step U2.

U5. [Inorder visit.] If traversing in inorder, visit NODE(Q).

U6. [Go to right.] If RLINK(Q) ≠ Λ and RTAG(Q) = "+", set Q ← RLINK(Q) and go to step U2.

U7. [Remove the thread.] Set R ← RLINK(Q), RTAG(Q) ← "+", RLINK(Q) ← Λ.

U8. [Go up.] Set Q ← R. Go back to step U5 if Q ≠ Λ, otherwise terminate the algorithm. ▮

Alternatively, the following slightly slower

1.562 amendments to Algorithm V — 10/17/79 — 71

steps V1 and V7: LOC(T) ↝ Λ
step V3: delete "(It is . . . .)"

1.562 the paragraph after Algorithm V — 3/25/81 — 72

line 2: to solve this problem ↝ to traverse in any of the three orders
line 6: 14.] ↝ 14.] A much simpler way to avoid the tag bits, at least for preorder and inorder traversal, was derived a few years later by J. M. Morris [*Information Proc. Letters* 9 (1979), 199–200]. See also the articles by G. Lindstrom . . . (etc., move the sentence from the end of the following paragraph to here)

1.562 new answer 22 (extends to page 563) — 10/17/79 — 73

22. Let rI4 ≡ R, rI5 ≡ Q, rI6 ≡ −P; use other conventions of Programs T and S.

```
U1 LD5  T              1      U1. Initialize. Q ← T.
   J5NZ U3             1
   JMP  DONE           0      Special exit if T = 0.
U4 ENN6 0,5            a − 1  U4. Insert a right thread. P ← Q.
   ENT5 0,4            a − 1  Q ← R.
4H ENT3 0,4            n − b  S ← R.
   LD4  1,3(RLINK)     n − b  R ← RLINK(S).
   J4NZ 4B             n − b  Repeat until R = Λ.
   ST6  1,3(RLINKT)    a − 1  RLINKT(S) ← −P.
U3 LD4  0,5(LLINK)     n      U3. Go to left. R ← LLINK(Q).
   J4NZ U4             n      To U4 if R ≠ Λ.
U5 JMP  VISIT          n      U5. Inorder visit.
U6 ENT4 0,5            n      U6. Go to right. R ← Q.
   LD5  1,5(RLINKT)    n      Q ← RLINKT(Q).
   J5P  U3             n      To U3 if Q > 0.
U7 STZ  1,5(RLINKT)    a      U7. Remove the thread.
U8 ENN5 0,5            a      U8. Go up. Q ← −Q.
   J5NZ U5             a      To U5 if Q ≠ Λ. ▮
```

Note that the search in step U4 is not time-consuming, since it examines each RLINK at most once. The total running time is $12n + 8a - 4b - 2$, where $n > 0$ is the number of nodes, a is the number of null RLINKs, and b is the number of nodes on the tree's "right path" T, RLINK(T), RLINK(RLINK(T)), etc. Thus, the algorithm is competitive with that of exercise 20. The running time of an analogous program based on Algorithm V of exercise 21 is $22n - 10$.

1.567 the missing MIX program on bottom four lines — 6/8/80 — 74

```
ST3  6F(0:2)
ST2  7F(0:2)
ENT2 8F
JMP  1F
```

1.568 program line 86 — 6/8/80 — 75

0,2 ↝ 0,2(RLINKT)

1.568 **improvements to program lines 93-100** 6/8/80 76

```
93   C4 LDA  0,1(LLINK)   C4. Anything to left?
94      JANZ 4B           Jump if LLINK(P) ≠ Λ.
95      STZ  0,2(LLINK)   LLINK(Q) ← Λ.
96   C5 LD2N 0,2(RLINKT)  C5. Advance. Q ← −RLINKT(Q).
97      LD1  0,1(RLINK)   P ← RLINK(P).
98      J2P  C5           Jump if RTAG(Q) was "−".
99      ENN2 0,2          Q ← −Q.
100  C6 J2NZ C2           C6. Test if complete.
```

1.568 **lines 3 and 4 of answer 14** 6/8/80 77

89–95, ... $18u$); $\leadsto$ 89–94, n; 95, $n-a$; 96–98, $n+1$; 99–100, $n-a$; 101–103, 1. The total time is $(36n+22)u$;

1.575 **exercise 12 line 5 (improves 1979 change #100)** 9/21/76 78

∞. $\leadsto$ ∞. Here $c(i,j)$ means $c(j,i)$ when $j<i$.

1.579 **in the biggest matrix** 5/1/79 79

change the label on row 3 and the label on column 3 from [10] to [20]

1.579 **in the second-biggest matrix, row 1** 5/1/79 80

a_{0m} $\leadsto$ a_{0n}

1.581 **new answer** 5/19/81 81

27. Let a_{ij} be the sum of $p(e)$ over all arcs e from V_i to V_j. We are to prove that $t_j=\sum_i a_{ij}t_i$ for all j. Since $\sum_i a_{ji}=1$, we must prove that $\sum_i a_{ji}t_j=\sum_i a_{ij}t_i$. But this is not difficult, because both sides of the identity represent the sum of all products $p(e_1)\ldots p(e_n)$ taken over subgraphs $\{e_1,\ldots,e_n\}$ of G such that $\mathrm{init}(e_i)=V_i$ and such that there is a unique oriented cycle contained in $\{e_1,\ldots,e_n\}$, where this cycle includes V_j. Removing any arc of the cycle yields an oriented tree; the lefthand side of the identity is obtained by factoring out the arcs that leave V_j, while the righthand side corresponds to those that enter V_j.

In a sense, this exercise is a combination of exercises 19 and 26.

1.582 **line -9** 3/1/79 82

Note: Kruskal's $\leadsto$ *Note:* Kruskal actually proved a stronger result, using a weaker form of embedding. His

1.582 **line -6** 3/25/81 83

305. $\leadsto$ 305. See N. Dershowitz, *Information Proc. Letters* **9** (1979), 212–215, for applications to termination of algorithms.

1.588 ***lines -4 and -3 of answer 32*** 3/16/81 84

is ... methods above $\leadsto$ is minimal. Still another proof, by G. Bergman, inductively replaces $d_k d_{k+1}$ by $(d_k+d_{k+1}-1)$ if $d_k>0$ [*Algebra Universalis* **8** (1978), 129–130].

The methods above

1.589 line 1 of answer 4 10/18/79 *85*

$l_j > l_{j+1}$ $\leadsto$ $l_j \ge l_{j+1}$

1.590 addendum to answer 10 10/18/79 *86*

(place the figure at the right margin and set the copy narrower, to its left)

The desired ternary tree is $\leadsto$

The desired ternary tree is shown at the right. F. K. Hwang has observed [*SIAM J. Appl. Math.* 37 (1979), 124–127] that a similar procedure is valid for minimum weighted path length trees having any prescribed multiset of degrees: at each step the smallest t weights are combined, where t is as small as possible.

1.590 new answers replacing answer 12 10/18/79 *87*

12. By exercise 9, it is the internal path length divided by n. [This holds for general trees as well.]

13. [Cf. J. van Leeuwen, *Proc. 3rd International Colloq. Automata, Languages, and Programming*, Edinburgh (July 1976), 382–410.]

H1. [Initialize.] Set $A[m-1+i] \leftarrow w_i$ for $1 \le i \le m$. Then set $x \leftarrow m$, $i \leftarrow m+1$, $j \leftarrow m-1$, $k \leftarrow m$. (During this algorithm $A[i] \le \cdots \le A[2m-1]$ is the queue of unused external weights and $A[k] \ge \cdots \ge A[j]$ is the queue of unused internal weights; the current left and right pointers are x and y.)

H2. [Find right pointer.] If $j < k$ or $A[i] \le A[j]$, set $y \leftarrow i$ and $i \leftarrow i+1$; otherwise set $y \leftarrow j$ and $j \leftarrow j-1$.

H3. [Create internal node.] Set $k \leftarrow k-1$, $L[k] \leftarrow x$, $R[k] \leftarrow y$, $A[k] \leftarrow A[x] + A[y]$.

H4. [Done?] Terminate the algorithm if $k = 1$.

H5. [Find left pointer.] (At this point $j \ge k$ and the queues contain a total of k unused weights. If $A[y] < 0$ we have $j = k$, $i = y+1$, and $A[i] > A[j]$.) If $A[i] \le A[j]$, set $x \leftarrow i$ and $i \leftarrow i+1$; otherwise set $x \leftarrow j$ and $j \leftarrow j-1$. Return to step H2. ∎

14. The proof for $k = m-1$ applies with little change. [Cf. *SIAM J. Appl. Math.* 21 (1971), 518.]

15. Use the combined-weight functions (a) $1 + \max(w_1, w_2)$ and (b) $x(w_1 + w_2)$, respectively, instead of $w_1 + w_2$ in (9). [Part (a) is due to M. C. Golumbic, *IEEE Trans.* C-25 (1976), 1164–1167; part (b) to T. C. Hu, D. Kleitman, and J. K. Tamaki, *SIAM J. Appl. Math.* 37 (1979), 246–256. Part (a) may be considered as the limiting case of part (b) as $x \to \infty$; Huffman's problem is, similarly, the limiting case as $x \to 1$, since $\sum (1+\epsilon)^{l_j} w_j = \sum w_j + \epsilon \sum w_j l_j + O(\epsilon^2)$.]

D. Stott Parker, Jr., has pointed out that a Huffman-like algorithm will also find the minimum of $w_1 x^{l_1} + \cdots + w_m x^{l_m}$ when $0 < x < 1$, if the two *maximum* weights are combined at each step. In particular, the minimum of $w_1 2^{-l_1} + \cdots + w_m 2^{-l_m}$, when $w_1 \le \cdots \le w_m$, is $w_1/2 + \cdots + w_{m-1}/2^{m-1} + w_m/2^{m-1}$.

16. Let $l_{m+1} = l'_{m+1} = 0$. Then

$$\sum_{1\le j\le m} w_j l_j \le \sum_{1\le j\le m} w_j l'_j = \sum_{1\le k\le m} (l'_j - l'_{j+1}) \sum_{1\le j\le k} w_j$$
$$\le \sum_{1\le k\le m} (l'_j - l'_{j+1}) \sum_{1\le j\le k} w'_j = \sum_{1\le j\le m} w'_j l'_j,$$

since $l'_j \ge l'_{j+1}$ as in exercise 4. The same proof holds for many other kinds of optimum trees, including those of exercise 10.

17. (a) This is exercise 14. (b) We can extend $f(n)$ to a concave function $f(x)$, so the stated inequality holds. Now $F(m)$ is the minimum of $\sum_{1\leq j<m} f(s_j)$, where the s_j are internal node weights of an extended binary tree on the weights 1, 1, ..., 1. Huffman's algorithm, which constructs the complete binary tree with $m-1$ internal nodes in this case, yields the optimum tree. Therefore the choice $k = 2^{\lceil \lg n/3 \rceil}$ yields the minimum in the recurrence, for each n. [*Reference: SIAM J. Appl. Math.* **31** (1976), 368–378. We can evaluate $F(n)$ in $O(\log n)$ steps; cf. exercises 5.2.3–20 and 21. If $f(n)$ is convex instead of concave, so that $\Delta^2 f(n) \geq 0$, the solution to the recurrence is obtained when $k = \lfloor n/2 \rfloor$.]

1.603 **new version of lines 22–24 (overrides previous changes)** 10/18/79 *88*

[This method is called the "LISP 2 garbage collector." An interesting alternative, which does not require the LINK field at the beginning of a node, can be based on the idea of linking together all pointers that point to each node—see Lars-Erik Thorelli, *BIT* **16** (1976), 426–441; F. Lockwood Morris, *CACM* **21** (1978), 662–665, **22** (1979), 571; and H. B. M. Jonkers, *Inf. Proc. Letters* **9** (1979), 26–30. Other methods have been published by B. K. Haddon and W. M. Waite, *Comp. J.* **10** (1967), 162–165; B. Wegbreit, *Comp. J.* **15** (1972), 204–208; D. A. Zave, *Inf. Proc. Letters* **3** (1975), 167–169.]

1.606 new answers 4/1/81 *89*

42. We can assume that $m \geq 6$. The main idea is to establish the occupancy pattern $R_{m-2}(F_{m-3}R_1)^k$ at the beginning of the memory, for $k = 0, 1, \ldots$, where R_j and F_j denote reserved and free blocks of size j. The transition from k to $k+1$ begins with

$$\begin{aligned} R_{m-2}(F_{m-3}R_1)^k &\to R_{m-2}(F_{m-3}R_1)^k R_{m-2}R_{m-2} \\ &\to R_{m-2}(F_{m-3}R_1)^{k-1} F_{2m-4}R_{m-2} \\ &\to R_{m-2}(F_{m-3}R_1)^{k-1} R_m R_{m-5} R_1 R_{m-2} \\ &\to R_{m-2}(F_{m-3}R_1)^{k-1} F_m R_{m-5} R_1; \end{aligned}$$

then the commutation sequence $F_{m-3}R_1F_mR_{m-5}R_1 \to F_{m-3}R_1R_{m-2}R_2R_{m-5}R_1 \to F_{2m-4}R_2R_{m-5}R_1 \to R_mR_{m-5}R_1R_2R_{m-5}R_1 \to F_mR_{m-5}R_1F_{m-3}R_1$ is used k times until we get $F_mR_{m-5}R_1(F_{m-3}R_1)^k \to F_{2m-5}R_1(F_{m-3}R_1)^k \to R_{m-2}(F_{m-3}R_1)^{k+1}$. Finally when k gets large enough there is an endgame that forces overflow unless the memory size is at least $(n - 4m + 11)(m - 2)$; details appear in *Comp. J.* **20** (1977), 242–244. [Note that the worst conceivable worst case, which begins with the pattern $F_{m-1}R_1F_{m-1}R_1F_{m-1}R_1\ldots$, is only slightly worse than this; the next-fit strategy of exercise 6 can produce this pattern.]

43. We will show that if $D_1, D_2, \ldots$ is any sequence of numbers such that $D_1/m + D_2/(m+1) + \cdots + D_m/(2m-1) \geq 1$ for all $m \geq 1$, and if $C_m = D_1/1 + D_2/2 + \cdots + D_m/m$, then $N_{FF}(n, m) \leq nC_m$. In particular, since

$$\frac{1}{m} + \frac{1}{m+1} + \cdots + \frac{1}{2m-1} = 1 - \frac{1}{2} + \cdots + \frac{1}{2m-3} - \frac{1}{2m-2} + \frac{1}{2m-1} > \ln 2,$$

the constant sequence $D_m = 1/(\ln 2)$ satisfies the necessary conditions. The proof is by induction on m. Let $N_j = nC_j$ for $j \geq 1$, and suppose that some request for a block of size m cannot be allocated in the leftmost N_m cells of memory. Then $m > 1$. For $0 \leq j < m$, we let N'_j denote the rightmost position allocated to blocks of sizes $\leq j$, or 0 if all reserved blocks are larger than j; by induction we have $N'_j \leq N_j$. Furthermore we let N'_m be the rightmost occupied position $\leq N_m$, so that $N'_m \geq N_m - m + 1$. Then the interval $(N'_{j-1}, N'_j]$ contains at least $\lceil j(N'_j - N'_{j-1})/(m+j-1)\rceil$ occupied cells, since its free blocks are of size $< m$ and its reserved blocks are of size $\geq j$. It follows that $n - m \geq$ number of occupied cells $\geq \sum_{1\leq j\leq m} j(N'_j - N'_{j-1})/(m+j-1) = mN'_m/(2m-1) - (m-1)\sum_{1\leq j<m} N'_j/(m+j)(m+j-1) > mN_m/(2m-1) - m - (m-1)\sum_{1\leq j<m} N_j(1/(m+j-1) - 1/(m+j)) = \sum_{1\leq j\leq m} nD_j/(m+j-1) - m \geq n - m$, a contradiction.

[This proof establishes slightly more than was asked. If we define the D's by $D_1/m + \cdots + D_m/(2m-1) = 1$, then the sequence $C_1, C_2, \ldots$ is $1, \frac{7}{4}, \frac{161}{72}, \frac{7483}{2880}, \ldots$; and the result can be improved further, even in the case $m = 2$, cf. exercise 38.]

1.617L entry for Abel, binomial formula generalized 3/16/81 *90*

398. $\rightsquigarrow$ 398, 501.

1.617L 9/4/79 *91*

al-Khowârizmî ... Mohammed $\rightsquigarrow$ al-Khwârizmî, abu Ja'far Muḥammad

1.618R entry for Best-fit 4/1/81 *92*

add p. 455

1.618R 3/16/81 *93*

Bergman, George Mark, 493, 588.

1.618R entry for Bernoulli polynomials — 10/25/79 — *94*
add p. 498

1.618R entry for Binary trees, complete — 3/15/81 — *95*
401. ⇝ 401, 590.

1.618R entry for Binary trees, copying of — 10/17/79 — *96*
332 ⇝ 331–332

1.618R entry for Binomial theorem, Abel's generalization — 3/16/81 — *97*
398. ⇝ 398, 501.

1.619L — 10/17/79 — *98*
Branislav, Ďurian, 562.

1.619R — 3/16/81 — *99*
Cauchy, Augustin Louis, 36–37, 501, 515, 578.

1.620L — 3/15/81 — *100*
Complete binary tree, 400–401, 590.

1.620L — 3/15/81 — *101*
Concave function, 405.

1.620L line -10 — 5/19/81 — *102*
strongly, 372, 377, 381.

1.620R — 3/15/81 — *103*
Convex function, 590.

1.620R entry for Copy a ... tree — 10/17/79 — *104*
332 ⇝ 331–332

1.621L — 3/25/81 — *105*
Dershowitz, Nachum, 582.

1.621R lines 3 and -21 — 7/8/81 — *106*
omit these entries about 'divided differences'

1.622R line 2 — 7/8/81 — *107*
add p. 472 to the Euler entry

1.622R — 10/18/79 — *108*
Farey, John, 157, 515.

1.623L entry for First-fit — 4/1/81 — *109*
add p. 455

1.623L 2/7/79 *110*

Fraenkel, Aviesri S., 248.

1.623L 2/7/79 *111*

Friedman, Daniel Paul, 420.

1.623R 3/15/81 *112*

Glassey, Charles Roger, 405.

1.623R 3/15/81 *113*

Golumbic, Martin Charles, 590.

1.623R 3/15/81 *114*

Hardy, Godfrey Harold, 12, 405, 490, 515.

1.623R 10/18/79 *115*

Haros, Ch., 515.

1.624L 3/15/81 *116*

Hu, Te Chiang, 405, 590.

1.624L 10/18/79 *117*

Hwang, Frank Kwangming, 405, 590.

1.625L 10/18/79 *118*

Jonkers, Henricus Bernardus Maria, 603.

1.625L 3/15/81 *119*

Karp, Richard Manning, 405.

1.625R 3/15/81 *120*

Kleitman, Daniel J., 590.

1.626R 3/15/81 *121*

Littlewood, John Edensor, 405.

1.627R 3/25/81 *122*

Morris, Francis Lockwood, 603.
Morris, Joseph Martin, 562.

1.627R 4/1/81 *123*

Next-fit method, 452 (exercise 6), 606.

1.628L 3/25/81 *124*

Olver, Frank William John, 499.

1.628R 5/19/81 *125*

Parker, Douglass Stott, Jr., 590.

1.629L *Pólya entry* 12/12/79 *126*
add p. 405

1.630L line 1 3/15/81 *127*
271, 458, 596. $\leadsto$ 271, 405, 458, 590, 596.

1.630L line 7 12/12/79 *128*
249. $\leadsto$ 249, 405, 590.

1.631L new subentry under Schröder 3/3/81 *129*
numbers, 534, 587.

1.631R 2/20/81 *130*
Spanning tree, minimum, 370–371.

1.631R 5/19/81 *131*
Steady states, 381.

1.631R 3/30/81 *132*
Steffensen, Johan Frederik, 498.

1.632L 5/19/81 *133*
Strongly connected directed graph, 372, 377, 381.

1.632R 3/15/81 *134*
Tamaki, Jeanne Keiko, 590.

1.633L *entry for Trees, copying of* 10/17/79 *135*
332 $\leadsto$ 331–332

1.633R 3/15/81 *136*
Tucker, Alan Curtiss, 405.

1.633R 3/15/81 *137*
van Leeuwen, Jan, 590.

1.634L 2/7/79 *138*
Wise, David Stephen, 420, 434, 595.

1.634R 3/1/79 *139*
Yao, Andrew Chi-Chih, 538.

1.634R 3/3/81 *140*
delete the entry for Benna Kay Young

2.xii line -6 4/12/81 *141*
300 $\leadsto$ 302

2.6 line -10 4/10/81 *142*

felt table $\leadsto$ felt-covered table

2.7 first line of exercise 7 3/6/81 *143*

least $\leadsto$ greatest

2.14 line 19 12/20/80 *144*

DEC 20 $\leadsto$ DECsystem 20

2.38 lines 14 and 17 1/27/81 *145*

too much space after 'Dr.' (twice)

2.45 line -9 1/27/81 *146*

though though $\leadsto$ though

2.55 line 10 2/2/81 *147*

0 and 1 $\leadsto$ 0 and n

2.58 exercise 19 1/27/81 *148*

Kolomogrov $\leadsto$ Kolmogorov

2.61 line -7 2/2/81 *149*

above the mean" and "runs below $\leadsto$ below the mean" and "runs above

2.64 line 4 after Algorithm P 9/9/80 *150*

exchange U_r, U_s $\leadsto$ exchange $U_r \leftrightarrow U_s$

2.66 left side of second equation in (14) 2/2/81 *151*

Z_{pj} $\leadsto$ Z_{qj}

2.67 right side of second equation in (18) 3/26/81 *152*

Z_{pj} $\leadsto$ Z_{qj}

2.68 big matrix display (22) 1/12/81 *153*

(I'll fix this so the numerators and denominators are a little bit further from the fraction lines)

2.75 line 4 2/2/81 *154*

$\alpha X_k + \beta Y_k$ $\leadsto$ $\alpha U_k + \beta V_k$

2.104 line -9 2/28/81 *155*

$6X_{n+2}$ $\leadsto$ $6X_{n+1}$

2.115 five lines before (3) — 156

(1976), $\rightsquigarrow$ (1977),

2.117 five lines after (10) — 3/2/81 — 157

28 (1958), 610; $\rightsquigarrow$ 29 (1958), 610–611;

2.125 line -2 — 4/28/81 — 158

$-4/\ln U$, $\rightsquigarrow$ $-4\ln U$,

2.127 Equation (28) — 4/13/81 — 159

$1/cu$ $\rightsquigarrow$ $1/(cu)$

2.129 three lines before (35) — 5/4/81 — 160

see G. Marsaglia, $\rightsquigarrow$ see E. B. Wilson and M. M. Hilferty, *Proc. Nat. Acad. Sci.* 17 (1931), 684–688; G. Marsaglia,

2.130 line -15 — 2/2/81 — 161

$\overline{(1-z)}$ $\rightsquigarrow$ $\overline{(1-Z)}$

2.135 line 2 — 4/13/81 — 162

$c\,g(t)$ $\rightsquigarrow$ $cg(t)$

2.136 line 19 — 5/4/81 — 163

(J. L. Bentley and J. D. Saxe.) Fint $\rightsquigarrow$ Find

2.142 line 1 — 1/18/81 — 164

3.5. $\rightsquigarrow$ *3.5.

2.143 line 16 — 165

$U_1, U_2,$ $\rightsquigarrow$ $U_0, U_1,$

2.164 line 4 — 2/2/81 — 166

defined in exercise 1.1–8.) $\rightsquigarrow$ discussed in Section 1.1.)

2.171 line -17 (and also page 172 line 12) — 4/10/81 — 167

`DIMENSION IA(1)` $\rightsquigarrow$ `DIMENSION IA(55)`

2.172 lines -3 to -5 of the FORTRAN subroutine — 12/12/80 — 168

`IRN55(IA)` $\rightsquigarrow$ `K = IRN55(IA)` (thrice)

2.184 line 1 — 9/26/80 — 169

l'Academie $\rightsquigarrow$ *l'Académie*

2.188 line -2 — 6/26/81 — *170*

l'Académie $\rightsquigarrow$ *l'Académie*

2.193 last line before exercises — 1/12/81 — *171*

roman $\rightsquigarrow$ Roman

2.195 last line of exercise 23 — 4/2/81 — *172*

zero. $\rightsquigarrow$ zero, if $0 \in D$. Show that this conclusion need not be true if $0 \notin D$.

2.198 Planck's constant replaces Dirac h — 1/10/81 — *173*

line 21: $\hbar = 1.0545$ $\rightsquigarrow$ $h = 6.6256$
line -3: $\hbar = (24, +.10545000)$. $\rightsquigarrow$ $h = (24, +.66256000)$.

2.201 step N5 — 1/12/81 — *174*

choose the ... odd. $\rightsquigarrow$ change f to the nearest multiple f' of b^{-p} such that $b^p f' + \frac{1}{2}b$ is odd.

2.210 line -4 — 1/12/81 — *175*

computer System, $\rightsquigarrow$ *Computer System,*

2.213 — 1/12/81 — *176*

move the two quotations down between exercise 19 and the beginning of 4.2.2

2.216 new (18) — 1/12/81 — *177*

$$|\delta(x)| = \frac{|\rho(x)|}{x} \le \frac{|\rho(x)|}{b^{e-1} + |\rho(x)|} \le \tfrac{1}{2}b^{e-p}/(b^{e-1} + \tfrac{1}{2}b^{e-p}) < \tfrac{1}{2}b^{1-p}.$$

2.218 line -2 — 4/27/81 — *178*

$(\epsilon_1 + \epsilon_2)$; $\rightsquigarrow$ $(\min(\epsilon_1, \epsilon_2))$;

2.222 lines 23–26 — 1/8/81 — *179*

line 23: but if $\rightsquigarrow$ if
line 24: occur. [Roy $\rightsquigarrow$ occur, although repeated rounding of a number like 2.5454 will lead to almost as much error. [Cf. Roy
line 25: On the other hand, since $\rightsquigarrow$ Some
line 26: remainder $\rightsquigarrow$ least significant digit
line 26: often. $\rightsquigarrow$ often. Exercise 23 demonstrates this advantage of round-to-even.

2.223 Planck's constant replaces Dirac h — 1/10/81 — *180*

line -17: $(-23, +.00010545)$ $\rightsquigarrow$ $(-23, +.00066256)$
line -16: $(-26, +.10545000)$ $\rightsquigarrow$ $(-26, +.66256000)$
line -10: $(0, +.00063507)$ $\rightsquigarrow$ $(1, +.00039903)$

2.225 replacement for bottom line 181

$h = [(-26, +.66252000), (-26, +.66261000)];$

2.226 replacement for line 3 1/10/81 182

$N \otimes h = [(-2, +.39898544), (-2, +.39907676)].$
(also change h to h on line 2)

2.227 corrections to bad German 4/22/81 183

line 24: *Begrund-* $\rightsquigarrow$ *Begrün-*
line 25: *ung der Rechenarithmetik* $\rightsquigarrow$ *dung der Rechnerarithmetik*

2.227 last line before exercises 6/16/81 184

1980 $\rightsquigarrow$ 1981

2.259 line 9 4/13/81 185

$\lfloor rAX/v_1 \rfloor.$ $\rightsquigarrow$ $\lfloor \mathrm{rAX}/v_1 \rfloor.$ $\rightsquigarrow$

2.268 exercise 36 1/15/81 186

Appendix B $\rightsquigarrow$ Appendix A

2.268 last line of exercise 36 4/28/81 187

1974.] $\rightsquigarrow$ 1973.]

2.276 line 21 4/30/81 188

Informací 3 $\rightsquigarrow$ *Informací (Information Processing Machines)* 3

2.305 line -5 1/18/81 189

q; $\rightsquigarrow$ 9;

2.307 last line of Example 1 1/27/81 190

$(14198757)_{10}.$ $\rightsquigarrow$ $(1419857)_{10}.$

2.314 new display for line 7 1/20/81 191

$$\frac{201}{3} \Big/ \left(\frac{66}{6} \cdot \frac{12}{3}\right) = 67/44.$$

2.353 line 4 2/4/81 192

partial fractions $\rightsquigarrow$ continued fractions

2.369 lines 1 and 2 2/23/81 193

4th ed. (Oxford, 1960) $\rightsquigarrow$ 5th ed. (Oxford, 1979)

2.371 line 7 6/16/81 194

+1. $\rightsquigarrow$ +1. [*Math. Comp.* 36 (1981), 627–630.]

2.374 line 6 — [illegible] — *195*

$S'[1,r]]_j$. ⇝ $S'[1,r]_j$.

2.377 top line of (16) — 1/26/81 — *196*

$x^{n_4 - -1}$ ⇝ $x^{n_4}-1$

2.377 line -5 — 2/1/81 — *197*

this, ⇝ this:

2.384 lines -7, -5, -4 — 1/11/81 — *198*

N ⇝ V (thrice)

2.384 last three lines — 6/16/81 — *199*

D. R. Hickerson ... 224. ⇝
H. C. Williams, *Math. Comp.* 36 (1981), 593–601.

2.385 line 25 — 3/25/81 — *200*

Dixon's method ⇝ Dixon's method [*Math. Comp.* 36 (1981), 255–260]

2.386 line -11 — 2/17/81 — *201*

1979 ⇝ 1978

2.388 line 12 — 2/12/81 — *202*

$\frac{1}{8}\ln p_1p_2 \approx 45$ ⇝ $\frac{1}{4}\ln p_1p_2 \approx 90$

2.388 line -16 — 4/22/81 — *203*

651 ⇝ 654

2.389 line 20 — 3/22/[illegible] — *204*

$\gcd(x,y)$ ⇝ $\gcd(x-y,N)$

2.391 first line of (23) — 1/27/81 — *205*

2203 2281, ⇝ 2203, 2281,

2.391 line 4 after (23) — 1/27/81 — *206*

CRAY-I ⇝ CRAY-1
see J. ⇝ see *Math. Comp.* 35 (1980), 1387–1390, and J.

2.396 line 2 — 3/31/81 — *207*

all primes ⇝ all odd primes

2.396 exercise 24 line 2 — 1/17/81 — *208*

$x = n$ ⇝ $x \bmod n = 0$

2.398 new exercise [illegible] *209*

39. [*HM30*] (L. Adleman.) Let p be a rather large prime number and let a be a primitive root modulo p; thus, all integers b in the range $1 \le b < p$ can be written $b = a^n \bmod p$, for some unique n with $1 \le n < p$.

Design an algorithm that almost always finds n, given b, in $O(p^\epsilon)$ steps for all $\epsilon > 0$, using ideas similar to those of Dixon's factoring algorithm. [*Hint:* Start by building a repertoire of numbers n_i such that $a^{n_i} \bmod p$ has only small prime factors.]

2.402 line 15 2/15/81 *210*

$r_1(x) = 0.$ $\leadsto$ $r_1(x) = r_2(x).$

2.402 line 2 of step D1 2/3/81 *211*

$\leftarrow$ $\leadsto$ $=$

2.407 line -2 3/3/81 *212*

$\gcd(v(x), \mathrm{pp}((r(x)))$ $\leadsto$ $\gcd(v(x), \mathrm{pp}(r(x)))$

2.409 fractions in (13) and (14) 4/28/81 *213*

(the numerators and denominators will be moved a bit further from the fraction lines)

2.414 line -4 6/5/81 *214*

(25) $\leadsto$ (26)

2.415 line 7 6/5/81 *215*

(16) and (17) $\leadsto$ (17) and (18)

2.429 line -5 2/2/81 *216*

$c < d$ $\leadsto$ $1 \le c < d$

2.430 line -10 2/22/81 *217*

$\gcd(g_d(x), t(x)^{(p^d-1)/2})$ $\leadsto$ $\gcd(g_d(x), t(x)^{(p^d-1)/2} - 1)$

2.430 line -4 6/16/81 *218*

Comp., to appear.] $\leadsto$ *Comp.* 36 (1981), 587–592.]

2.432 line -9 6/11/81 *219*

$(x^2 - 13 - 7)$ $\leadsto$ $(x^2 - 13x - 7)$

2.432 line -8 3/3/81 *220*

are factors $\leadsto$ could be a factor

2.433 bottom line 5/21/81 *221*

$d > \frac{1}{2}r.$ $\leadsto$ $d \le \frac{1}{2}r.$

2.434 line 11 — 4/27/.. — *222*

2^{r-1} ⇝ $2^{r-1}-1$

2.438 line 3 of exercise 18 — 4/27/81 — *223*

$\cdots u_0u_n^{n-1}$. ⇝ $\cdots + u_0u_n^{n-1}$.

2.439 line 14 — 3/3/81 — *224*

mod 2 ⇝ modulo 2

2.442 three lines before Algorithm A — 6/18/81 — *225*

5 ⇝ .5

2.482 line 16 — 1/10/81 — *226*

Math., to appear. ⇝ *Math.* 7 (1981), 73–125.]

2.484 bottom line — 5/6/81 — *227*

$2n^2+2$ ⇝ $2n^2+2n$

2.487 the display after (46) — 1/27/81 — *228*

$\begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}$ ⇝. $\begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}$

2.496 line 26 — 1/10/81 — *229*

462. ⇝ 462; *JACM* 27 (1980), 822–830. See also his interesting discussion of commutative bilinear forms in *SIAM J. Computing* 9 (1980), 713–728.

2.506 lines 4-5 — 4/29/81 — *230*

their quotient, etc., ⇝ and sometimes their quotient,

2.517 line -12 — 2/2/81 — *231*

$X_0 = a$ ⇝ $X_1 = a$

2.519 line 2 — 5/5/81 — *232*

$6\sqrt{\pi/2m}$ ⇝ $\sqrt{\pi/2m}$

2.520 exercise 15 — 2/1/81 — *233*

$(m-1)^m/m$, ⇝ $(m-1)^m/m^m$,

2.523 lines 8 and 9 — 2/2/81 — *234*

so … result. ⇝ so $(a^{2^{e-1}}-1)/(a-1) \equiv 0$ (modulo 2^e) iff $(a^{2^{e-1}}-1)/2 \equiv 0$ (modulo $2^{e+1}/2$), which is true.

2.523 line 4 of exercise 11 — 2/2/81 — *235*

$(\pm x)^{2e-f-1}$ $\leadsto$ $(\pm x)^{2^e-f-1}$
x^{2e-f} $\leadsto$ x^{2^e-f}
$(\pm x)^{2e-f}$ $\leadsto$ $(\pm x)^{2^e-f}$

2.531 line -2 — 2/2/81 — *236*

$F_n(x) - F_n(y)$, $\leadsto$ $F_n(y) - F_n(x)$,

2.536 exercise 15 — :/2/81 — *237*

and S has $\leadsto$ and X has

2.536 line -5 — 2/2/81 — *238*

$\begin{pmatrix} U_1' \, U_2' \ldots U_{n-1}' \\ V_1' \, V_2' \ldots V_{n-1}' \end{pmatrix}$ $\leadsto$ $\begin{pmatrix} U_0' \, U_1' \ldots U_{n-1}' \\ V_0' \, V_1' \ldots V_{n-1}' \end{pmatrix}$

2.540 line 3 — 2/2/81 — *239*

$\left(\left(\frac{a(x+c_0/d)}{m/d}\right)\right)$ $\leadsto$ $\left(\left(\frac{a(x+c_0/d)}{m/d}\right)\right)$

2.543 line 5 of exercise 5 — 2/2/81 — *240*

$(h' - qh)^2$ $\leadsto$ $(h' - q'h)^2$

2.546 line 2 of exercise 24 — 2/2/81 — *241*

mod n $\leadsto$ mod m

2.547 line 10 of exercise 27 — 2/2/81 — *242*

r_t $\leadsto$ u_t

2.550 line -2 of answer 10 — 4/4/81 — *243*

b_1, $\leadsto$ $(b_1$,

2.550 first line of answer 11 — 4/9/81 — *244*

$\int_0^x$ $\leadsto$ $\int_3^x$

2.554 lines 2 and 3 — 5/4/81 — *245*

[*ACM* ... appear.] $\leadsto$ [This technique was apparently introduced in the 1960s by David Seneschol; cf. *Amer. Statistician* 26,4 (October 1972), 56–57. The alternative of generating n uniform numbers and sorting them is probably faster unless n is rather large, but this method is particularly valuable if only a few of the largest or smallest X's are desired. Note that $(F^{-1}(X_1), \ldots, F^{-1}(X_n))$ will be sorted deviates having distribution F.]

2.561 bottom line of answer 37 — 3/20/81 — *246*

334.] $\leadsto$ 334; see also the Ph.D. thesis of Thomas N. Hersog, Univ. of Maryland (1975).]

2.565 **answer 23** 3/21/81 *247*

line 4: zero since it is $\leadsto$ zero if $0 \in D$, since T is
line 5: 10^k $\leadsto$ b^k
line 6: zero. $\leadsto$ zero. On the other hand, as pointed out by K. A. Brakke, every real number has infinitely many representations in the number system of exercise 21.
line 9: less $\leadsto$ fewer

2.568 **line 14** 6/15/81 *248*

$k_T(z)$. $\leadsto$ $k_T(z)$. [Cf. *J. Algorithms* **2** (1981), 31–43.]

2.568 **replacement for previous answer** 1/10/81 *249*

1. $N = (62, +.60\ 22\ 52\ 00)$; $h = (37, +.66\ 25\ 60\ 00)$. Note that $10h$ would be $(38, +.06\ 62\ 56\ 00)$.

2.570 **line 9** 1/15/81 *250*

after this instruction 'ENT2 0', insert a new one 'JXNZ *+3' on a new line

2.570 **line 2 of answer 19** 1/15/81 *251*

$b/20$ $\leadsto$ $b/2\ 0$

2.573 ***new answer 23*** 2/2/81 *252*

23. If $u \geq 0$ or $u \leq -1$ we have $u \bmod 1 = u \bmod 1$ (the first with circled mod), so the identity holds. If $-1 < u < 0$, then u (mod) $1 = u \oplus 1 = u + 1 + r$ where $|r| \leq \frac{1}{2}b^{-p}$; the identity holds iff $\text{round}(1 + r) = 1$, so it always holds if we round to even. With the text's rounding rule the identity fails iff b is a multiple of 4 and $-1 < u < 0$ and $u \bmod 2b^{-p} = \frac{3}{2}b^{-p}$ (e.g., $p = 3$, $b = 8$, $u = -(.0124)_8$).

2.589 ***line 7*** 11/11/80 *253*

, to appear. $\leadsto$ **9** (1980), 490–508.

2.596 **answer 20** 5/21/81 *254*

$p(\ldots)$ $\leadsto$ $(\ldots)p$ (thrice)

2.608 **line -1** 11/11/80 *255*

2.4771 is chosen "optimally" as the root of $(p^2 - 1)\ln p = p^2 - p + 1$. See *BIT* **20** (1980), 176–184.]

2.613 **exercise 24** 1/17/81 *256*

line 3: passes $\leadsto$ fails
lines 4 and 5: at most $\frac{1}{4}qn + \ldots < \frac{1}{2}N$ $\leadsto$
at most $-1 + q(b_n + 1) + \min(b_n + 1, r) \leq$
$q\bigl(\frac{1}{4}(n-1) + 1\bigr) + \min\bigl(\frac{1}{4}(n-1), r - 1\bigr) <$
$\frac{1}{3}qn + \min(\frac{1}{4}n, r) = \frac{1}{3}N + \min(\frac{1}{4}n - \frac{1}{3}r, \frac{2}{3}r) \leq \frac{1}{3}N + \frac{1}{8}n \leq \frac{1}{2}N$

2.614 **last three lines of exercise 27** 12/12/80 *257*

$n = 1, 3, 7, 13, 15, 25, 39, 55, 75, 85, 127, 1947, 3313, 4687, 5947$. See R. M. Robinson, *Proc. Amer. Math. Soc.* **9** (1958), 673–681; G. V. Cormack and H. C. Williams, *Math. Comp.* **35** (1980), 1419–1421.]

2.616 **new answer** 4/5/81 *258*

39. After finding $a^{n_i} \bmod p = \prod_{1\le j\le m} p_j^{e_{ij}}$ for enough n_i, we can solve $\sum_i x_{ijk}e_{ij} + (p-1)t_{jk} = \delta_{jk}$ in integers x_{ijk}, t_{jk} for $1 \le j,k \le m$ (e.g., as in 4.5.2–23), thereby knowing the solutions $N_j = (\sum_i x_{ijk}e_{jk}) \bmod (p-1)$ to $a^{N_j} \bmod p = p_j$. Then if $ba^{n'} \bmod p = \prod_{1\le j\le m} p_j^{e'_j}$, we have $n + n' \equiv \sum_{1\le j\le m} e'_j N_j$ (modulo p). [Cf. *Proc. IEEE Symp. Foundations of Comp. Sci.* **20** (1979), 55–60.]

2.619 **last line of exercise 12** 1/10/81 *259*

[*JACM*, to appear.] ⤳ [Cf. *JACM* **27** (1980), 701–717.]

2.626 **last line of exercise 19** 4/27/81 *260*

u_0. ⤳ u_0. [The idea of this proof actually goes back to T. Schönemann, *J. für die reine ... Math.* **32** (1846), 100.]

2.637 **line -14** 12/1/80 *261*

D. J. S. Brown ⤳ D. J. Spencer Brown

2.637 **end of answer 26** 5/21/81 *262*

190.] ⤳ 190.] In fact, as Richard Brent has observed, the number of operations can be reduced to $O(d^2 \log n)$, or even to $O(d \log d \log n)$ using exercise 4.7–6, if we first compute $x^n \bmod (x^d - a_1x^{d-1} - \cdots - a_d)$ and then replace x^j by x_j.

2.639 **line 8 of answer 39** 6/15/81 *263*

arcs. ⤳ arcs. [Cf. *J. Algorithms* **2** (1981), 13–21.]

2.639 **exercise 41** 1/27/81 *264*

NP hard ⤳ NP-hard
NP complete ⤳ NP-complete (twice)

2.647 **line 6 of exercise 41** 2/9/81 *265*

(1960), ⤳ (1971),

2.653 **line 8** 3/7/81 *266*

x_{2m-1} ⤳ $x_{2m-1}u^{m-1}$

2.653 **first line of step N2** 3/7/81 *267*

$x_{mj+i}\, Y_{ij}$ ⤳ $x_{mj+i},\, Y_{ij}$

2.657 **last two lines of exercise 13** 12/13/80 *268*

Fred ... (1979). ⤳ Richard P. Brent, Fred G. Gustavson, and David Y. Y. Yun, *J. Algorithms* **1** (1980), 259–295.

2.666 line -4 3/3/81 269

Σ ↝ 4Σ

2.668R 4/5/81 270

Adleman, Leonard Max, 380, 386, 396, 398.

2.669R 4/2/81 271

Balanced decimal number system, 195, 565.

2.670L 5/4/81 272

delete the entry for Jon Bentley

2.670L Berlekamp entry 3/3/81 273

420, 423, ↝ 420–423,

2.670R 4/2/81 274

Brakke, Kenneth Allen, 565.

2.670R Richard Brent entry 5/21/81 275

add p. 637

2.670R 3/2/81 276

Brooks, Frederick Phillips, Jr., 210.

2.670R 12/1/80 277

delete 'Brown, D. J. Spencer, 637.'

2.671R near the Congruential sequence entry 1/12/81 278

delete the spurious comma in the right margin

2.672L 12/12/80 279

Cormack, Gordon Villy, 614.

2.672L 1/27/81 280

CRAY-1, 391.

2.672L 12/20/80 281

DECsystem 20, 14.

2.673R 4/5/81 282

Dixon, John Douglas, 356, 385, 395, 397, 398.

2.675R 4/13/81 283

Galois, Evariste, ↝ Galois, Évariste,

2.676L GRH entry 3/12/81 284

Reimann ↝ Riemann

2.676R 3/20/81 *285*
Hersog, Thomas Nelson, 166, 558, 561.

2.676R 6/16/81 *286*
delete the entry for D. R. Hickerson

2.676R 5/4/81 *287*
Hilferty, Margaret M., 129.

2.677R **entry for Knuth, Donald** 3/2/81 *288*
vi–vii, ⇝ iv, vi–vii,

2.678L **Leibniz entry** 5/22/81 *289*
freiherr ⇝ Freiherr

2.678R **new subentry under Logarithm** 4/5/81 *290*
modulo p, 398.

2.678R 2/2/81 *291*
Mandelbrot, Benoît Baruch, 564.

2.680R **line -24** 4/2/81 *292*
balanced decimal, 195, 565.

2.680R 1/27/81 *293*
NP-complete problem, 480, 550, 639.

2.682L 3/3/81 *294*
Pippenger, Nicholas John, 461, 639.

2.682L 12/12/80 *295*
delete 'Plass, Michael Frederick, 614.'

2.683L **entry for Primitive root** 4/5/81 *296*
add p. 398

2.684R **entry for Rounding** 2/2/81 *297*
364. ⇝ 364, 573.

2.684R 5/4/81 *298*
delete the entry for James Saxe

2.685L 4/27/81 *299*
Schönemann, Theodor, 626.

2.685L 5/4/81 *300*
Seneschol, David, 554.

2.685L **Shanks entry** 6/16/81 *301*

384, 385, ⇝ 385,

2.685R 3/25/81 *302*

Sobol', Il'ia Meerovich, 519.

2.685R 12/1/80 *303*

Spencer Brown, David John, 637.

2.687R **von Mises entry** 5/22/81 *304*

edler ⇝ Edler

2.688L 6/16/81 *305*

Williams, Hugh Cowie, 378, 384, 397, 614.

2.688L 5/4/81 *306*

Wilson, Edwin Bidwell, 129.

2.688L 12/1/80 *307*

Wynn-Williams, Charles Eryl, 186.

2.688R **Zaremba entry** 1/20/81 *308*

Slanislaw ⇝ Stanislaw

3.9 **exercise 17** 1/31/79 *309*

How ⇝ (This n is called the *index* of b modulo p, with respect to a.) How

3.10 **line -9** 7/4/81 *310*

less ⇝ fewer

3.19 **second line of exercise 9** 4/13/81 *311*

its own inverse ⇝ an involution (i.e., its own inverse)

3.23 **lines 17 and 22** 10/18/79 *312*

Anuyogadvarā ⇝ *Anuyogadvāra* (twice)

3.76 **line -7** 10/18/79 *313*

$p(n)$ ⇝ $p(N)$

3.90 **caption** 10/18/79 *314*

Fig. 12 ⇝ Fig. 12.

3.108 **line -14** 2/7/79 *315*

betweeen ⇝ between

3.204 **lines -12 and -11** 6/24/80 *316*

This proof … 6.) $\leadsto$ The reader may have noticed a pattern in the three formulas just proved; Paul Stockmeyer and Frances Yao have shown that the pattern holds in general, i.e., that the lower bounds derived by the strategy above suffice to establish the values $M(m, m+d) = 2m + d - 1$ for $m \geq 2d - 2$. [*SIAM J. Computing* 9 (1980), 85–90.]

3.317 **correction to step B1** 11/14/79 *317*

transpose the two sentences 'Then write …' $\leftrightarrow$ 'Set $A[0,0]$ …'

3.321 **line 4** 10/5/79 *318*

individual $\leadsto$ individually

3.378 **new exercise** 10/10/80 *319*

19. [*HM25*] (R. W. Floyd.) Show that the lower bound of Theorem F can be improved to

$$\frac{(k+1)nb\lg b + nb/\ln 2}{b+c}\left(1 + O\left(\frac{\log b}{b}\right)\right)$$

when $n = b^k$, for fixed k as $b \to \infty$, and also to $nb + O(n/\log n)$ for fixed b as $n \to \infty$, in the sense that some initial configuration must require at least this many stops. [*Hint:* Count the configurations that can be sorted after s stops.]

3.381 **the line for "Diminishing increments"** 3/17/81 *320*

$15N^{1.25}$ $\leadsto$ $15N^{1.25} + 10\log_3(N/3)$

3.384 **line 15** 3/15/81 *321*

is an incidental remark which appears in an article $\leadsto$ is in a book by Robert Feindler, *Das Hollerith-Lochkarten-Verfahren* (Berlin: Reimar Hobbing, 1929), 126–130; it was also mentioned at about the same time in an article

3.389 **line -11 (also make this change throughout the book)** 3/25/81 *322*

data base $\leadsto$ *database*

3.392 **lines -12 and -11** 10/10/80 *323*

Cincinnati Redlegs $\leadsto$ Chicago White Sox

3.405 **line 3 of exercise 19** 6/1/81 *324*

i, j? $\leadsto$ $i \neq j$?

3.412 **line -6** 4/8/81 *325*

$\left\lfloor\frac{N+2^{j-1}}{2^j}\right\rfloor = \left(\frac{N}{2^j}\right)$ rounded, $\leadsto$ $\left\lfloor\frac{N+2^{j-1}}{2^j}\right\rfloor$,

3.419 **line 22** 6/2/80 326

but ... 23). ⇝ but a successful search will require about one more iteration, on the average, because of (2). Since the inner loop is performed only about $\lg N$ times, this tradeoff between an extra iteration and a faster loop does not save time unless N is extremely large. (See exercise 23.) On the other hand Bottenbruch's algorithm will find the rightmost occurrence of a given key when the table contains duplicates, and this property is occasionally important.

3.420 **line -9** 3/2/81 327

11 ⇝ **11.**

3.422 **line 9** 6/2/80 328

necessary!) ⇝ necessary on a successful search!)

3.422 **exercise 27 line 6** 1/24/79 329

n ⇝ k

3.439 **update to 1979 change #240** 2/28/81 330

the Hu-Kleitman-Tamaki paper appeared in *SIAM J. Appl. Math.* **37** (1979), 246–256

3.448 **last line of exercise 6** 4/13/81 331

of C'_{n-1}? ⇝ of this distribution?

3.449 **exercise 23 (cf. 1979 change #311)** 11/15/78 332

$p_1 = 5$ ⇝ $p_1 = 9$

3.451 **line -3** 3/20/81 333

Akademiia ⇝ *Akademii*

3.471 **insert quotation before Section 6.2.4** 3/15/81 334

Samuel considered the nation of Israel, tribe by tribe,
and the tribe of Benjamin was picked by lot.
Then he considered the tribe of Benjamin, family by family,
and the family of Matri was picked by lot.
Then he considered the family of Matri, man by man,
and Saul son of Kish was picked by lot.
But when they looked for Saul he could not be found.

—1 Samuel 10:20–21

3.472 **line 11** 1/31/79 335

$\log_2$ ⇝ lg

3.476 **clarifications** 1/31/79 *336*

line -14: new node ⇝ new key
line -11: nodes ⇝ internal nodes
line -10: nodes ⇝ internal nodes
line -8: a node ⇝ a node while building a tree of N keys

3.480 **exercise 5** 2/23/79 *337*

flowing.") ⇝ flowing"; pass up the key that makes the remaining two parts most nearly equal in size.)

3.491 **Figure 33** 2/23/79 *338*

(It would be desirable to show the 5-bit binary codes in fine print under the TEXT line; to make room, "TEXT:" should be brought up to a line by itself. Furthermore, this figure needs to be redrawn; the word in node γ should be changed to (THE), and the word in node ϵ should be changed to (THAT); also, the dotted line at the lower left of node ϵ should become a circular dotted line that points right back to node ϵ (cf. β and ς), while the dotted line at the lower right of ϵ should point up to γ.)

3.491 **line -12** 2/23/79 *339*

contains the number 24 (the ⇝ would contain the number 24 (which indicates the

3.491 **line -10** 2/23/79 *340*

$\log_2$ ⇝ lg

3.492 **replacement for lines 2 through 11** 12/27/79 *341*

A search in Patricia's tree is carried out as follows: Suppose we are looking up the word THE (bit pattern 10111 01000 00101). We start by looking at the SKIP field of the root node α, which tells us to examine the first bit of the argument. It is 1, so we move to the right. The SKIP field in the next node, γ, tells us to look at the $1 + 11 = 12$th bit of the argument. It is 0, so we move to the left. The SKIP field of the next node, ϵ, tells us to look at the $(12 + 1)$st bit, which is 1; now we find $\text{RTAG} = 1$, so we go back to node γ, which refers us to the TEXT. The search path we have taken would occur for any argument whose bit pattern is 1xxxx xxxxx x01..., and we must check to see if it matches the unique key beginning with that pattern.

3.506 **line 8** 1/24/79 *342*

Section ⇝ Sections

3.507 **update to 1979 change #259** 3/1/79 *343*

850 ⇝ 850, 22 (1979), 104,

3.518 **corrected analysis** 1/10/80 *344*

line 9, a new equation: $C'_N = 1 + \dfrac{N(N-1)}{2M^2} \approx 1 + \frac{1}{2}\alpha^2$

line 6 after (19): The method introduces a tag bit in each entry; the average number of probes needed in an unsuccessful search therefore decreases slightly, from (18) to

$$\left(1-\frac{1}{M}\right)^N + \frac{N}{M} \approx e^{-\alpha} + \alpha. \qquad (18')$$

line 8 after (19): delete the sentence 'If separate ... $\alpha > 1$.'

line 11 after (19): $\frac{1}{2}$. $\leadsto$ $\frac{1}{2}$. However, it is usually preferable to use an alternative scheme that puts the first colliding elements into an auxiliary storage area, allowing lists to coalesce only when this auxiliary area has filled up; see exercise 43.

3.519 **bottom line** 6/6/80 *345*

$9u$ $\leadsto$ $8u$

3.522 **last line of (24)** 4/4/80 *346*

`ORR` $\leadsto$ `OR`

3.524 **several refinements** 1/10/80 *347*

line 1 of (30): `-M-1,1` $\leadsto$ `1-M,1`

line 1 just after (30): In this $\leadsto$

Program D takes a total of $8C + 19A + B + 26 - 13S - 17S1$ units of time; modification (30) saves about $15(A - S1) \approx 7.5\alpha$ of these in a successful search. In this

furthermore, Fig. 42 needs to be more accurately redrawn using the following data:

$\alpha =$	0.0	0.2	0.4	0.6	0.8	0.9	0.92	0.94	0.96	0.98	0.99
$L =$	24.0	24.9	26.3	29.3	38.0	55.5	64.3				
$D =$	23.0	25.7	28.8	32.6	38.4	43.9	45.7	47.9	51.2	56.8	62.5
$D_{\text{mod}} =$	23.0	24.2	26.0	28.8	34.1	39.6	41.5	43.9	47.2	53.1	58.9

3.526 **new paragraph after line 19** 1/1/81 *348*

E. G. Mallach [*Comp. J.* **20** (1977), 137–140] has experimented with refinements of Brent's variation, and further results have been obtained by Gaston H. Gonnet and J. Ian Munro [*SIAM J. Computing* **8** (1979), 463–478].

3.539 **Change to curves S and SO in Figure 44(a)** 1/10/80 *349*

$\alpha =$	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
$S =$	1.0	1.005	1.020	1.045	1.080	1.125	1.180	1.245	1.320	1.405	1.500
$SO =$	1.0	1.003	1.013	1.029	1.051	1.079	1.112	1.151	1.195	1.244	1.299

3.543 **new rating for exercise 10** 3/1/79 *350*

[*M43*] $\leadsto$ [*M38*]

3.544 exercise 14 (replacement for lines 3 and following) 2/23/79 *351*

2-bit TAG field and two link fields called LINK and AUX, with the following interpretation:

TAG(P) = 0 indicates a word in the list of available space; LINK(P) points to the next entry in this list, and AUX(P) is unused.

TAG(P) = 1 indicates any word in use where P is not the hash address of any key in the scatter table; the other fields of the word in location P may have any desired format.

TAG(P) = 2 indicates that P is the hash address of at least one key; AUX(P) points to a linked list specifying all such keys, and LINK(P) points to another word in the list memory. Whenever a word with TAG(P) = 2 is accessed during the processing of any list, it is necessary to set P ← LINK(P) repeatedly until reaching a word with TAG(P) ≤ 1. (For efficiency we might also then change prior links so that it will not be necessary to skip over the same scatter table entries again and again.)

Show how to define suitable algorithms for inserting and retrieving keys in a combined table of this sort.

3.544 exercise 23 2/23/79 *352*

[*23*] ⇝ [*33*]

3.546 replacements for exercises 34(c), 35, 36 1/10/80 *353*

(c) Express the average number of probes for a successful search in terms of this generating function. (d) Deduce the average number of probes in an *unsuccessful* search, considering variants of the data structure in which the following conventions are used: (i) hashing is always to a list head (cf. Fig. 38); (ii) hashing is to a table position (cf. Fig. 40), but all keys except the first of a list go into a separate overflow area; (iii) hashing is to a table position and all entries appear in the hash table.

35. [*M24*] Continuing exercise 34, what is the average number of probes in an unsuccessful search when the individual lists are kept in order by their key values? Consider data structures (i), (ii), and (iii).

36. [*M23*] Continuing exercise 34(d), find the *variance* of the number of probes when the search is unsuccessful, using data structures (i) and (ii).

3.546 new wording of exercises 37 and 40 1/10/80 *354*

▶ **37.** [*M29*] Eq. (19) gives the average number of probes in separate chaining when the search is successful; what is the *variance* of this quantity?

40. [*M33*] Eq. (15) gives the average number of probes used by Algorithm C in an unsuccessful search; what is the *variance* of this quantity?

3.546 new wording for exercise 39 (keep the old last line) 6/1/80 *355*

39. [*M27*] Let $c_N(k)$ be the total number of lists of length k formed when Algorithm C is applied to all M^N hash sequences (35). Find a recurrence relation on the numbers $c_N(k)$ that makes it possible to determine a simple formula for the sum

$$S_N = \sum_k \binom{k}{2} c_N(k).$$

3.546 **New rating for exercise 43** 8/8/80 *356*

[*M42*] $\leadsto$ [*HM44*]

3.563 **line 12** 6/10/80 *357*

{NEEDLE, NODDLE, NOODLE} $\leadsto$ {NEEDLE, NIDDLE, NODDLE, NOODLE, NUDDLE}

3.576 **addendum to 1976 change #324** 4/5/81 *358*

John M. Pollard [*Math. Comp.* 32 (1978), 918–924] has discovered an elegant way to solve this problem with very little memory in about $O(\sqrt{p})$ steps, based on the theory of random mappings. See also the asymptotically faster method of exercise 4.5.4–39.

3.593 **display in answer 25** 2/26/80 *359*

$z^n/n!$ $\leadsto$ z^n

3.608 **line -8** 2/15/79 *360*

$z^{N+1-\delta_{s1}}$ $\leadsto$ z^{N+1}

3.609 **answers 24 and 27** 3/1/79 *361*

line 3 of answer 24: replace by lines 8 and 9 of answer 27

lines 8 and 9 of answer 27 should be:

$\alpha \neq \beta$; $g(x) = x^\beta(\ln x + C)$ for $\alpha = \beta$. We have $p_t(-t-2) = 0$; so the general solution to our differential equation is

3.614 **line -6 of answer 55** 1/29/80 *362*

rA $\leadsto$ rA

3.617 **line -6** 12/14/79 *363*

(exercise 4.5.4–8) is a $O(N)$ $\leadsto$
(as implemented in exercise 4.5.4–8) is a $O(N \log\log N)$

3.619 **answer 31** 3/16/81 *364*

lines 1 and 2: Let ... $B[i]$ for $\leadsto$ (Solution by J. Edighoffer.) Let A be an array of $2n$ elements such that $A[2\lfloor i/2\rfloor] \leq A[2i]$ and $A[2\lfloor i/2\rfloor - 1] \geq A[2i-1]$ for $1 < i \leq n$; furthermore we require that $A[2i-1] \geq A[2i]$ for

line 4: twin-heap $\leadsto$ twin heap

3.624 **line -5** 5/1/79 *365*

$g^{n+1}_{M,N}(z)$ $\leadsto$ $g^{(n+1)}_{M,N}(z)$

3.633 **new answer** 11/11/80 *366*

14. [*SIAM J. Computing* 9 (1980), 298–320.]

3.665 **new answer** 10/10/80 *367*

19. There are at least $(nb)!/b!^{2n}$ configurations, and the number that can be obtained from a given one after s stops is at most $\bigl((n-1)\binom{b+c}{b}\bigr)^s$, which is less than $n^s 2^{(b+c)s}$. Hence $s > (\ln(nb)! - 2n\ln b!)/(\ln n + (b+c)\ln 2)$ and the stated results follow.

3.667 answer 19 6/1/81 368

line 1: We $\rightsquigarrow$ Assuming that $d(i,i) = 0$, we
line 3: is due $\rightsquigarrow$ for $i \neq j$ is due

3.672 line 4 3/15/81 369

[From exercise 6.2.1–25b we can therefore $\rightsquigarrow$ [By exercise 6.2.1–25(b) we can use the mean and variance of C'_n to

3.672 line 1 of answer 15 10/23/79 370

a_i $\rightsquigarrow$ a_j

3.675 answer 11 (improvement to 1979 change #312) 1/31/79 371

produces $\rightsquigarrow$ results in (twice)
[To be published.] $\rightsquigarrow$ [*SIAM J. Computing* 8 (1979), 33–41.]

3.680 addendum to 1976 change #359 3/25/81 372

suffice.] $\rightsquigarrow$ suffice. In general, if we want to compress n sparse tables containing respectively $x_1, \ldots, x_n$ nonzero entries, a 'first-fit' method that offsets the jth table by the minimum amount r_j that will not conflict with the previously placed tables will have $r_j \leq (x_1 + \cdots + x_{j-1})x_j$, since each previous nonzero entry can block at most x_j offsets. This worst-case estimate gives $r_j \leq 93$ for the data in Table 1, guaranteeing that any twelve tables of length 30 containing respectively 10, 5, 4, 3, 3, 3, 3, 3, 2, 2, 2, 2 nonzero entries can be packed into $93 + 30$ consecutive locations regardless of the pattern of the nonzeros. Further refinements of this method have been developed by R. E. Tarjan and A. C. Yao, *CACM* 22 (1979), 606–611.]

3.683 answer 14 line 4 1/31/79 373

TAG $\rightsquigarrow$ `TAG`

3.688 new answer 10 3/1/79 374

10. See F. M. Liang's elegant proof in *Discrete Math.* 28 (1979), 325–326.

3.689 line 2 3/16/81 375

lists, $\rightsquigarrow$ lists, following a suggestion of Allen Newell,

3.689 new paragraph inserted at beginning of answer 14 2/23/79 376

14. According to the stated conventions, the notation "`X` ⇐ `AVAIL`" of 2.2.3–6 now stands for the following operations: "Set `X` ← `AVAIL`; then set `X` ← `LINK(X)` zero or more times until either `X` = 0 (an `OVERFLOW` error) or `TAG(X)` = 0; finally set `AVAIL` ← `LINK(X)`."

3.689 new paragraph appended at end of answer 14 2/23/79 377

Another way to place a hash table "on top of" a large linked memory, using coalescing lists instead of separate chaining, has been suggested by J. S. Vitter [Ph.D. thesis, Stanford Univ. (1980), 72–73].

3.690 **new answer 23** 6/6/80 *378*

23. J. S. Vitter [Ph.D. thesis, Stanford Univ. (1980), 61–68] has introduced a deletion method for coalesced chaining that preserves the distribution of search times.

3.693 **answer 34** 1/10/80 *379*

lines 4 and 5: C'_N ... all keys. $\rightsquigarrow$ Consider the total number of probes to find all keys, not counting the fetching of the pointer in the list head table of Fig. 38 if such a table is used.

line -1: Thus we obtain (18), (19). $\rightsquigarrow$ (d) In case (i) a list of length k requires k probes (not counting the list-head fetch), while in case (ii) it requires $1 + \delta_{k0}$. Thus in case (ii) we get $C'_N = \sum(k + \delta_{k0})P_{Nk} = P'_N(1) + P_N(0) = N/M + (1 - 1/M)^N \approx \alpha + e^{-\alpha}$, while case (i) has simply $C'_N = N/M = \alpha$. The formula $MC'_N = M - N + NC_N$ applies in case (iii), since $M - N$ hash addresses will discover an empty table position while N will cause searching to the end of some list; this yields (18).

3.693 **new answer 35** 1/10/80 *380*

35. (i) $\sum(1 + \frac{1}{2}k - (k+1)^{-1})P_{Nk} = 1 + N/2M - M(1 - (1 - 1/M)^{N+1})/(N+1) \approx 1 + \frac{1}{2}\alpha - (1 - e^{-\alpha})/\alpha$. (ii) Add $\sum \delta_{k0}P_{Nk} = (1 - 1/M)^N \approx e^{-\alpha}$ to the result of (i). (iii) Assume that when an unsuccessful search begins at the jth element of a list of length k, the given key has random order with respect to the other k elements, so the expected length of search is $(j \cdot 1 + 2 + \cdots + (k + 1 - j) + (k + 1 - j))/(k + 1)$. Summing on j now gives $MC'_N = M - N + M\sum(k^3 + 9k^2 + 2k)P_{Nk}/6(k+1) = M - N + M(\frac{1}{6}N(N-1)/M^2 + \frac{3}{2}N/M - 1 + (M/(N+1))(1 - (1 - 1/M)^{N+1}))$; hence $C'_N \approx \frac{1}{2}\alpha + \frac{1}{6}\alpha^2 + (1 - e^{-\alpha})/\alpha$.

3.693 **answer 36** 6/6/80 *381*

line 1, replace first sentence by: (i) $N/M - N/M^2$. (ii) $\sum(\delta_{k0} + k)^2 P_{Nk} = \sum(\delta_{k0} + k^2)P_{Nk} = P_N(0) + P''_N(1) + P'_N(1)$.

line -1, add new remark: [For data structure (iii), a more complicated analysis like that in exercise 37 would be necessary.]

3.694 **replacement for lines 1-3 and big display of answer 39** 6/1/80 *382*

39. (This approach to the analysis of Algorithm C was suggested by J. S. Vitter.) We have $c_{N+1}(k) = (M - k)c_N(k) + (k - 1)c_N(k - 1)$ for $k \geq 2$, and furthermore $\sum kc_N(k) = NM^N$. Hence $S_{N+1} = \sum_{k\geq 2}\binom{k}{2}c_{N+1}(k) = \sum_{k\geq 2}\binom{k}{2}((M-k)c_N(k) + (k-1)c_N(k-1)) = \sum_{k\geq 1}((M+2)\binom{k}{2} + k)c_N(k) = (M+2)S_N + NM^N$.

3.694 **line 1 of answer 40** 6/1/80 *383*

$\binom{j}{2}$ replaced by $\binom{j+1}{3}$. $\rightsquigarrow$ $\binom{k}{2}$ replaced by $\binom{k+1}{3}$.

3.694 **new answer** 6/6/80 *384*

43. Let $N = \alpha M'$ and $M = \beta M'$, and let $e^{-\lambda} + \lambda = 1/\beta$, $\rho = \alpha/\beta$. Then $C_N \approx 1 + \frac{1}{2}\rho$ and $C'_N \approx \rho + e^{-\rho}$, if $\rho \leq \lambda$; $C_N \approx \frac{1}{8\rho}(e^{2(\rho-\lambda)} - 1 - 2(\rho - \lambda))(3 - 2/\beta + 2\lambda) + \frac{1}{4}(\rho + \lambda) + \frac{1}{4}\lambda(1 - \lambda/\rho)$ and $C'_N \approx 1/\beta + \frac{1}{4}(e^{2(\rho-\lambda)} - 1)(3 - 2/\beta + 2\lambda) - \frac{1}{2}(\rho - \lambda)$, if $\rho \geq \lambda$. For $\alpha = 1$ we get the smallest $C_N \approx 1.69$ when $\beta \approx .853$; the smallest $C'_N \approx 1.79$ occurs when $\beta \approx .782$. So it pays to put the first collisions into an area that doesn't conflict with hash addresses, even though a smaller range of hash addresses causes more collisions to occur. These results are due to Jeffrey S. Vitter [Ph.D. thesis, Stanford Univ. (1980); *Proc. Symp. Foundations Comp. Sci.* **21** (1980), 238–247].

3.710R 10/18/79 *385*

Anuyogadvarā ⇝ Anuyogadvāra

3.712L **delete 1979 change #334** 3/1/79 *386*

(Fan Chung no longer mentioned on page 688)

3.713L 3/16/81 *387*

Edighoffer, Judy Lynn Harkness, 619.

3.713R 3/15/81 *388*

Feindler, Robert, *384*.

3.714L 3/25/81 *389*

First-fit allocation, 471, 680.

3.715L 4/5/81 *390*

Index modulo *p*, 9.

3.716R 3/1/79 *391*

Liang, Franklin Mark, 688.

3.718L 3/15/81 *392*

Newell, Allen, 689.

3.718R **(this entry now moves to the preceding column)** 4/4/80 *393*

ORR ⇝ OR

3.719L **Vaughan Pratt entry** 1/24/79 *394*

add p. 450

3.720L 3/15/81 *395*

Samuel, 471.

3.720L 3/25/81 *396*

Sparse array, 680.

3.721L **update to 1979 change #384** 3/1/79 *397*

Sprugnoli, Renso, 507.

3.721R 6/24/80 *398*

Stockmeyer, Paul Kelly, 204.

3.721R 3/25/81 *399*

Tarjan, Robert Endre, 216, 624, 680.

3.722L 3/16/81 *400*

Twin heap, 619.

3.722R 1/10/81 *401*

Vitter, Jeffrey Scott, 689, 690, 694.

3.722R **von Mises entry** 5/22/81 *402*

edler ⇝ Edler

3.722R 3/25/81 *403*

Yao, Andrew Chi-Chih, 232, 235, 422, 479, 549, 639, 678, 680.
Yao, Foong Frances, 204, 232, 422.