〔付記〕 このプログラムでは a_i を毎回新たに全部を並べかえているが，どこまですませたかを記録しておけば，多少能率を上げることもできる．

最後の CHANGE の部分は，5020 では前記の形でなく，じっさいには下記のような算法によった．⊕は bitwise addition (exclusive or) の演算である．この中の for の文は，いずれも 5020 の variable length 命令の repeat 機能を活用することにより，それぞれ 1 命令ですませえた．しかしこのままの形を JUSE ALGOL で書くと(たとえ⊕のサブルーチンを工夫してはさんでも)，さらに能率がおちるので，この方は前記の形にかきかえて実行した．

```
CHANGE: begin integer P;
        W:=N-J;Q:=W+2;
        if W≠Q* 2 then P:=A[N-Q];
        for I:=1 step 1 until W do
          A[J+I]:=A[J+I]⊕A[N+1-I];
        for I:=1 step 1 until Q do
          A[J+I]:=A[J+I]⊕A[N+1-I];
        if W≠Q* 2 then A[N-Q]:=P
        end;
```

6506. 辞書式順列 I

和田英一（東京大学工学部計数工学科）
浦部雍子（日本科学技術研修所）

0,1,2,……, $N-1$ の N 個の数字でつくった $N!$ 個の順列を辞書式にならべたときの M ($\leqq 0 M<N!$) 番目の順列 $P_M=(P_0, P_1, P_2, \cdots\cdots, P_{N-1})$ の P_I ($I=0,1,2,\cdots\cdots, N-1$) をもとめる．

まず M を $M=A_0\cdot 0!+A_1\cdot 1!+\cdots+A_{N-1}\cdot(N-1)!$ ただし $A_0=0$, $0\leqq A_1\leqq 1$, $0\leqq A_2\leqq 2, \cdots\cdots$, $0\leqq !A_{N-1}\leqq N-1$ とあらわす A_I をもとめ，添字の小さい方の A_I から順に，自分より大きい添字をもっている A_J について，添字の小さい方から順に比較し，それが自分に等しいか，自分より小さいとき，自分の数を一つふやすという操作をくりかえす．こうしてできた A の配列を反対にすると P_I がもとまる．$M\geqq N!$ のとき ERROR≠0 となり P_M はもとまらない．

ただし M は，この procedure を実行するとこわれる．以下のプログラムは ALGOLIP (OKITAC 5090 のための ALGOL) によるものである．array の上限が 99 になっているが，実際に必要な array の size は P[0: N−1], A[0: N−1] である．

(Beckenbach, E.F. 編 Applied Combinatorial Mathematics p. 21 参照)

```
begin integer N,M, ERROR;
  integer array P[0:99];
  procedure MTHPERMUTATION;
    begin integer I, J, Q;
    integer array A[0: 99];
      for I:=0 step 1 until N-1 do
        begin Q:=M÷(I+1);
              A[I]:=M-Q* (I+1);
              M:=Q
        end;
      ERROR:=M;
      if M=0 then
      for I:=0 step 1 until N-1 do
        begin
          for J:=I+1 step 1 until N-1 do
            if A[I]≧A[J] then A[I]:=A[I]+1;
          P[N-1-I]:=A[I]
        end
      end;
    comment driver program;
    begin integer I, PI;
      N:=READINTEGER;
L1:   M:=READINTEGER;
      PRINTINTEGER(M);
      MTHPERMUTATION;
      if ERROR=0 then
        begin
          PI:=0;
          for I:=0 step 1 until N-1 do
              PI:=PI* 10+P[I];
          PRINTINTEGER(PI)
        end
      else PRINTSTRING (' ERROR');
      CRLF (1); go to L1
  end
end
```

6507. 辞書式順列 II

和田英一（東京大学工学部計数工学科）
浦部雍子（日本科学技術研修所）

辞書式順列 I の逆である．P_I ($I=0, 1, \cdots\cdots, N-1$) であたえられた順列 (0, 1, 2, ……, $N-1$ よりなる) が辞書式順序にならべたとき，なん番目になるかを，NUMBER におく．もとの順列が正しければ ERROR =0 で，NUMBER がもとまり，正しくなければ，

ERROR=1 で，NUMBER はもとまらない．実際に必要な array の size は P[0: N-1]，A[0: N-1] である．

```
begin integer N, NUMBER, ERROR;
  integer array P[0: 99];
  procedure PERMUTATIONNUMBER;
    begin integer I, J; integer array A[0: 99];
      ERROR: =0;
      for I: =N-1 step-1 until 0 do
    begin A[I]:=P[N-1-I];
      for J: =N-1 step -1 until I+1 do
    begin if A[I]=A[J] then ERROR: =1;
          if A[I]>A[J] then A[I]: =A[I]-1
    end;
    if A[I]>I then ERROR: =1
    end;
      if ERROR=0 then begin
      NUMBER: =0;
      for I: =N-1 step -1 until 0 do
    NUMBER: =NUMBER*(I+1)+A[I] end
    end;
comment driver program;
    begin integer I, PI, Q;
      N: =READINTEGER;
L 1: PI: =READINTEGER;
    PRINTINTEGER(PI);
    for I: =N-1 step -1 until 0 do
      begin
        Q: =PI÷10;
        P[I]: =PI-Q*10;
        PI: =Q
      end;
    PERMUTATIONNUMBER;
  if ERROR=0 then
    PRINTINTEGER (NUMBER)
 else PRINTSTRING (' ERROR');
    CRLF(1); go to L 1
    end
  end
```