2. GAUTSCHI, W. Computational aspects of three-term recurrence relations. Unpublished.
3. ——. Algorithm 221—Gamma function. *Comm. ACM 7* (Mar. 1964), 143.
4. HERNDON, J. R. Algorithm 62—A set of associate Legendre polynomials of the second kind. *Comm. ACM 4* (July 1961), 320–321; Remark on Algorithm 62. *Comm. ACM 4* (Dec. 1961), 544.
5. *NBS Tables of Associated Legendre Functions*. Columbia University Press, New York, 1945.
6. RIEGELS, F. Formeln und Tabellen für ein in der räumlichen Potentialtheorie auftretendes elliptisches Integral. *Archiv der Mathematik 2* (1949/50), 117–125.
7. SIEKMANN, J. Concerning an integral occurring in airfoil theory. *SIAM Review 3* (1961), 243–246.
8. ——. Analysis of ring aerofoils of elliptic cross section, Part I: General theory. *J. SIAM 11* (1963), 941–963.
9. ——. Note on a Riegels-type integral. *Z. Angew. Math. Phys. 15* (1964), 79–83.
10. ŽURINA, M. I., AND KARMAZINA, L. N. Tablicy funkciĭ Ležandra $P_{-\frac{1}{2}+i\tau}(x)$, Vol. I. Akad. Nauk SSSR, Moscow, 1962.
11. ——, AND ——. Tablicy funkciĭ Ležandra $P^{1}_{-\frac{1}{2}+i\tau}(x)$. Akad. Nauk SSSR, Moscow, 1963.

ALGORITHM 260
6-J SYMBOLS [Z]

J. H. GUNN (Recd. 13 Nov. 1964)
Nordisk Institut for Teoretisk Atomfysik, Copenhagen, Denmark

```
real procedure SJS (J1, J2, J3, L1, L2, L3, factorial);
  value J1, J2, J3, L1, L2, L3;
  integer J1, J2, J3, L1, L2, L3;
  array factorial;
```

comment *SJS* calculates the 6-*j* symbols defined by the following formula

$$\begin{Bmatrix} j1 & j2 & j3 \\ l1 & l2 & l3 \end{Bmatrix} = \Delta(j1,j2,j3)\Delta(j1,l2,l3)\Delta(l1,j2,l3)\Delta(l1,l2,j3) \times \textstyle\sum_z (-1)^z (z+1)!/((z-j1-j2-j3)!(z-j1-l2-l3)!\,(z-l1-j2-l3)!(z-l1-l2-j3)!(j1+j2+l1+l2-z)!\,(j2+j3+l2+l3-z)!(j3+j1+l3+l1-z)!)$$

where

$$\Delta(a,b,c) = \left[\frac{(a+b-c)!\,(a-b+c)!\,(-a+b+c)!}{(a+b+c+1)!}\right]^{\frac{1}{2}}$$

and where $j1 = J1/2$, $j2 = J2/2$, $j3 = J3/2$, $l1 = L1/2$, $l2 = L2/2$, $l3 = L3/2$. [Reference formula 6.3.7 page 99 of EDMONDS, A. R. Angular momentum in quantum mechanics. In *Investigations in Physics, 4*, Princeton U. Press, 1957]. The parameters of the procedure *J1, J2, J3, L1, L2, L3* are interpreted as being twice their physical value, so that actual parameters may be inserted as integers. Thus to calculate the 6-*j* symbol

$$\begin{Bmatrix} 2 & 2 & 0 \\ 2 & 2 & 0 \end{Bmatrix}$$

the call would be *SJS* (4, 4, 0, 4, 4, 0, *factorial*). The procedure checks that the triangle conditions for the existence of a coefficient are satisfied and that $j1 + j2 + j3$, $j1 + l2 + l3$, $l1 + j2 + l3$ and $l1 + l2 + j3$ are integral. If the conditions are not satisfied the value of the procedure is zero. The parameter *factorial* is an array containing the factorials from 0 up to at least 1 + largest of $j1 + j2 + j3$, $j1 + l2 + l3$, $l1 + j2 + l3$ and $l1 + l2 + j3$. Since in actual calculations the procedure *SJS* will be called many times it is more economical to have the factorials in a global array rather than compute them on every entry to the procedure. The notation is consistent with that used in the procedure for calculating Vector-coupling coefficients. See Algorithm 252, Vector Coupling or Clebsch-Gordan Coefficients [*Comm. ACM 8* (Apr. 1965), 217];

```
begin integer w, wmin, wmax;
real omega;
real procedure delta (a, b, c);
  value a, b, c;
  integer a, b, c;
begin delta := sqrt (factorial [(a+b−c)÷2]
  × factorial [(a−b+c)÷2]
  × factorial [(−a+b+c)÷2]/factorial [(a+b+c+2)÷2])
end delta;
if J1 + J2 < J3 ∨ abs(J1 − J2) > J3 ∨ J1 + J2 + J3 ≠
  2 × ((J1+J2+J3)÷2)
∨ J1 + L2 < L3 ∨ abs(J1−L2) > L3 ∨ J1 + L2 + L3 ≠ 2 ×
  ((J1+L2+L3)÷2)
∨ L1 + J2 < L3 ∨ abs(L1−J2) > L3 ∨ L1 + J2 + L3 ≠ 2 ×
  ((L1+J2+L3)÷2)
∨ L1 + L2 < J3 ∨ abs(L1−L2) > J3 ∨ L1 + L2 + J3 ≠ 2 ×
  ((L1+L2+J3)÷2)
then SJS := 0 else
begin
  omega := 0;
  wmin := J1 + J2 + J3;
  if wmin < J1 + L2 + L3 then wmin := J1 + L2 + L3;
  if wmin < L1 + J2 + L3 then wmin := L1 + J2 + L3;
  if wmin < L1 + L2 + J3 then wmin := L1 + L2 + J3;
  wmax := J1 + J2 + L1 + L2;
  if wmax > J2 + J3 + L2 + L3 then wmax := J2 + J3 +
    L2 + L3;
  if wmax > J3 + J1 + L3 + L1 then wmax := J3 + J1 +
    L3 + L1;
  for w := wmin step 2 until wmax do
  omega := omega + (if w=4×(w÷4) then 1 else −1)
    × factorial [w÷2+1]/(factorial [(w−J1−J2−J3)÷2]
    × factorial [(w−J1−L2−L3)÷2]
    × factorial [(w−L1−J2−L3)÷2]
    × factorial [(w−L1−L2−J3)÷2]
    × factorial [(J1+J2+L1+L2−w)÷2]
    × factorial [(J2+J3+L2+L3−w)÷2]
    × factorial [(J3+J1+L3+L1−w)÷2]);
  SJS := delta (J1, J2, J3) × delta (J1, L2, L3)
    × delta (L1, J2, L3) × delta (L1, L2, J3) × omega;
end
end SJS
```

ALGORITHM 261
9-J SYMBOLS [Z]

J. H. GUNN (Recd. 13 Nov. 1964)
Nordisk Institut for Teoretisk Atomfysik, Copenhagen, Denmark

```
real procedure NJS(J11, J12, J13, J21, J22, J23, J31, J32, J33,
    factorial);
  value J11, J12, J13, J21, J22, J23, J31, J32, J33;
  integer J11, J12, J13, J21, J22, J23, J31, J32, J33;
  array factorial;
```

comment *NJS* calculates the 9-*j* symbols defined by the following formula

$$\begin{Bmatrix} j11 & j12 & j13 \\ j21 & j22 & j23 \\ j31 & j32 & j33 \end{Bmatrix} = \textstyle\sum_k (-1)^{2k}(2k+1) \begin{Bmatrix} j11 & j21 & j31 \\ j32 & j33 & k \end{Bmatrix} \begin{Bmatrix} j12 & j22 & j32 \\ j21 & k & j23 \end{Bmatrix} \begin{Bmatrix} j13 & j23 & j33 \\ k & j11 & j12 \end{Bmatrix}.$$

where $j11 = J11/2$, $j12 = J12/2$, $j13 = J13/2$, $j21 = J21/2$,

$j22 = J22/2$, $j23 = J23/2$, $j31 = J31/2$, $j32 = J32/2$, $j33 = J33/2$ [Reference formula 6.4.3 page 101 of EDMONDS, A. R. Angular momentum in quantum mechanics. In *Investigations in Physics*, *4*, Princeton U. Press, 1957]. The parameters of the procedure $J11$, $J12$, $J13$, $J21$, $J22$, $J23$, $J31$, $J32$, $J33$ are interpreted as being twice their physical value, so that actual parameters may be inserted as integers. Thus to calculate the 9-*j* symbol

$$\begin{Bmatrix} 2 & 2 & 0 \\ 2 & 2 & 0 \\ 0 & 0 & 0 \end{Bmatrix}$$

the call would be *NJS* (4, 4, 0, 4, 4, 0, 0, 0, 0, *factorial*). The procedure checks that the triangle conditions for the existence of a coefficient are satisfied and that $j11 + j21 + j31$, $j21 + j22 + j23$, $j31 + j32 + j33$, $j11 + j12 + j13$, $j12 + j22 + j32$, $j13 + j23 + j33$ are integral. If the conditions are not satisfied the value of the procedure is zero. The parameter *factorial* is an array containing the factorials from 0 up to at least 1 + largest of $j11 + j21 + j31$, $j21 + j22 + j23$, $j31 + j32 + j33$, $j11 + j12 + j13$, $j12 + j22 + j32$, $j13 + j23 + j33$. The procedure makes use of the procedure *SJS* [Algorithm 260, 6-*j* symbols, *Comm. ACM 8* (Aug. 1965), 492], for calculating 6-*j* symbols;

```
begin integer k, kmin, kmax;
  real NJ;
  if J11 + J21 < J31 ∨ abs(J11−J21) > J31 ∨ J11 + J21 +
    J31 ≠ 2 × ((J11+J21+J31)÷2)
  ∨ J21 + J22 < J23 ∨ abs(J21−J22) > J23 ∨ J21 + J22 +
    J23 ≠ 2 × ((J21+J22+J23)÷2)
  ∨ J31 + J32 < J33 ∨ abs(J31−J32) > J33 ∨ J31 + J32 +
    J33 ≠ 2 × ((J31+J32+J33)÷2)
  ∨ J11 + J12 < J13 ∨ abs(J11−J12) > J13 ∨ J11 + J12 +
    J13 ≠ 2 × ((J11+J12+J13)÷2)
  ∨ J12 + J22 < J32 ∨ abs(J12−J22) > J32 ∨ J12 + J22 +
    J32 ≠ 2 × ((J12+J22+J32)÷2)
  ∨ J13 + J23 < J33 ∨ abs(J13−J23) > J33 ∨ J13 + J23 +
    J33 ≠ 2 × ((J13+J23+J33)÷2)
  then NJS := 0 else
  begin NJ := 0;
    kmin := abs(J21−J32);
    if kmin < abs(J11−J33) then kmin := abs(J11−J33);
    if kmin < abs(J12−J23) then kmin := abs(J12−J23);
      kmax := J21 + J32;
    if kmax > J11 + J33 then kmax := J11 + J33;
    if kmax > J12 + J23 then kmax := J12 + J23;
      for k := kmin step 2 until kmax do
    NJ := NJ + (if k=2×(k÷2) then 1 else −1) × (k+1) ×
      SJS(J11, J21, J31, J32, J33, k, factorial) ×
      SJS(J12, J22, J32, J21, k, J23, factorial) ×
      SJS(J13, J23, J33, k, J11, J12, factorial);
    NJS := NJ
  end
end NJS
```

ALGORITHM 262
NUMBER OF RESTRICTED PARTITIONS OF N [A1]

J. K. S. McKAY (Recd. 7 Dec. 1964 and 9 Mar. 1965)
Computer Unit, University of Edinburgh, Scotland

procedure *set* (*p*, *N*); **integer** *N*; **integer array** *p*;
comment The number of partitions of *n* with parts less than or equal to *m* is set in *p*[*n*, *m*] for all *n*, *m* such that $N \geq n \geq m \geq 0$.

REFERENCES:

1. GUPTA, H., GWYTHER, C. E., AND MILLER, J. C. P. Tables of partitions. In *Royal Society Mathematical Tables, vol. 4*, Cambridge U. Press, 1958.
2. HARDY, G. H., AND WRIGHT, E. M. *The Theory of Numbers.* Ch. 19, 4th ed., Clarendon Press, Oxford, 1960;

```
begin integer m, n;
  p[0, 0] := 1;
  for n := 1 step 1 until N do
  begin p[n, 0] := 0;
    for m := 1 step 1 until n do
      p[n, m] := p[n, m−1] +
        p[n−m, if n−m<m then n−m else m]
  end
end set
```

ALGORITHM 263
PARTITION GENERATOR [A1]

J. K. S. McKAY (Recd. 7 Dec. 1964 and 9 Mar. 1965)
Computer Unit, University of Edinburgh, Scotland.

procedure *generate* (*p*, *N*, *position*, *ptn*, *length*);
integer array *p*, *ptn*; **integer** *N*, *length*, *position*;
comment The partitions of *N* may be mapped in their natural order, 1 − 1, onto the consecutive integers from 0 to $P(N)-1$ where $P(N)(=p[N, N])$ is the number of unrestricted partitions of *N*. The array *p* is set by the procedure *set* [Algorithm 262, Number of Restricted Partitions of *N*, *Comm. ACM 8* (Aug. 1965), 493]. On entry *position* contains the integer into which the partition is mapped. On exit *length* contains the number of parts and *ptn*[1: *length*] contains the parts of the partition in descending order.

REFERENCE:

1. LITTLEWOOD, D. E. *The Theory of Group Characters.* Ch. 5, 2nd ed., Clarendon Press, Oxford, 1958;

```
begin integer m, n, psn;
  n := N;  psn := position;  length := 0;
A:  length := length + 1;  m := 1;
B:  if p[n, m] < psn then begin m := m + 1;  go to B end else
    if p[n, m] > psn then
C:  begin
      ptn[length] := m;  psn := psn − p[n, m−1];  n := n − m;
      if n ≠ 0 then go to A;  go to D
    end
    else m := m + 1;  go to C;
D:  end generate
```

ALGORITHM 264
MAP OF PARTITIONS INTO INTEGERS [A1]

J. K. S. McKAY (Recd. 7 Dec. 1964 and 9 Mar. 1965)
Computer Unit, University of Edinburgh, Scotland

integer procedure *place*(*p*, *n*, *ptn*); **value** *n*;
integer array *p*, *ptn*; **integer** *n*;
comment *place* is the inverse of the procedure *generate* [Algorithm 263, Partition Generator, *Comm. ACM 8* (Aug. 1965), 493]. The array *p* is set by the procedure *set* [Algorithm 262, Number of Restricted Partitions of *N*, *Comm. ACM 8* (Aug. 1965), 493]. The procedure produces the integer into which the partition of *n*, stored in descending order of parts in *ptn*[1] onwards, is mapped;

```
begin integer j, d;
  d := 0;
  if n = 0 then go to B;
  j := 0;
A: j := j + 1;  d := p[n, ptn[j]−1] + d;  n := n − ptn[j];
  if n ≠ 0 then go to A;
B: place := d
end place
```