```
> restart;
```

Maple 12

The procedure W3j is derived from the C program described by William J. Thompson.
William J. Thompson, Angular Momentum: An Illustrated Guide to Rotational Symmetries for Physical Systems, John Wiley and Sons Inc., 1994

Addition of Four Angular Momenta

Using Racah's formula to determine Wigner 3j coeficients

$$\begin{bmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & m_3 \end{bmatrix} = (-1)^{j_1 - j_2 - m_3} \cdot \mathrm{N} \cdot S$$

$$\mathrm{N} = \sqrt{\frac{(j3+j1-j2)!(j3-j1+j2)!(j1+j2-j3)!(j3-m3)!(j3+m3)!}{(j1+j2+j3+1)!(j1-m1)!(j1+m1)!(j2-m2)!(j2+m2)!}}$$

$$\mathrm{S} = \sum_{kmin}^{kmax} \frac{(-1)^{(kmin+j2+m2)}(j2+j3+m1-k)!(j1-m1+k)!}{(k)!(j3-j1+j2-k)!(j3-m3-k)!(k+j1-j2+m3)!}$$

where
kmin = maximun of [0 **and** $j2-j1-m3$] **and** kmax = minimum of [$j3-j1+j2$ **and** $j3-m3$]

Clebsch-Gordan Coefficients, C-G, from Wigner 3j Coefficients

$$\langle j_1,j_2;m_1,m_2|j_1,j_2;j_3,m_3\rangle = (-1)^{-j_1+j_2-m_3}\sqrt{2j_3+1}\begin{bmatrix} j_1 & j_2 & j_3 \\ m_1 & m_2 & -m_3 \end{bmatrix}$$

**** **Note that we are using the following conventions** ****

coefficient	$\{a,b,c,m_a,m_b,m_c\} \equiv \langle a,b;m_a,m_b|a,b;c,m_c\rangle$
C-G value	$[\pm N] \equiv \pm\sqrt{N}$

This procedure determines the Wigner 3j coefficient using Racah's formula

> W3j :=**proc**(j1, j2, j3, m1, m2, m3)

local *k*, *kmin*, *kmax*, *N*, *S*, *P*;

$kmin := \max(0, \mathrm{j2} - \mathrm{j1} - \mathrm{m3});$

$kmax := \min(\mathrm{j3} - \mathrm{j1} + \mathrm{j2}, \mathrm{j3} - \mathrm{m3});$

$P := (-1)^{(\mathrm{j1} - \mathrm{j2} - \mathrm{m3})};$

$$N := \sqrt{\frac{(\mathrm{j3} + \mathrm{j1} - \mathrm{j2})!(\mathrm{j3} - \mathrm{j1} + \mathrm{j2})!(\mathrm{j1} + \mathrm{j2} - \mathrm{j3})!(\mathrm{j3} - \mathrm{m3})!(\mathrm{j3} + \mathrm{m3})!}{(\mathrm{j1} + \mathrm{j2} + \mathrm{j3} + 1)!(\mathrm{j1} - \mathrm{m1})!(\mathrm{j1} + \mathrm{m1})!(\mathrm{j2} - \mathrm{m2})!(\mathrm{j2} + \mathrm{m2})!}};$$

$$S := \sum_{k=kmin}^{kmax} \frac{(-1)^{(k + \mathrm{j2} + \mathrm{m2})} \cdot (\mathrm{j2} + \mathrm{j3} + \mathrm{m1} - k)!(\mathrm{j1} - \mathrm{m1} + k)!}{(k)!(\mathrm{j3} - \mathrm{j1} + \mathrm{j2} - k)!(\mathrm{j3} - \mathrm{m3} - k)!(k + \mathrm{j1} - \mathrm{j2} + \mathrm{m3})!};$$

return $(P \cdot N \cdot S)$;

end proc:

This procedure returns a signed squared coefficient.

$\pm a\sqrt{n} = \pm a^2 n = \pm$ **N**

> SQ :=**proc**(*n*)

local *sign*, *c*;

if (*evalf*(*n*) < 0) **then** *sign* := -1 **else** *sign* := 1 **end if**;

c := *simplify*((*n*·*n*)·*sign*);

return (*c*);

end proc:

This procedure determines the value of the C-G coefficients; e.g., ⟨j1,j2;m1,m2|j1,j2;j12,m12⟩

⟨j1,j2;m1,m2|j1,j2;j12,m12⟩ = [± N]

[± N] ≡ ± $\sqrt{N}$

```
> VCo := proc(j1, j2, j3, j4, j12, j34, J, M, v)
      local m1, m2, m3, m4, m12, m34, c1, c2, c3, s;
      s := " = ";
      if v = 0 then printf("|%a,%a,%a,%a;%a,%a,%a,%a>\n", j1, j2, j3, j4, j12, j34, J, M) end if;
      for m12 from -j12 by 1 to j12 do
        for m34 from -j34 by 1 to j34 do
          for m1 from -j1 by 1 to j1 do
            for m2 from -j2 by 1 to j2 do
              for m3 from -j3 by 1 to j3 do
                for m4 from -j4 by 1 to j4 do
                  if (m1 + m2) = m12 and (m3 + m4) = m34 and (m12 + m34) = M then
  # selection rule
                    c1 := (-1)^(-j1 + j2 - m12)·sqrt(2·j12 + 1)· W3j(j1, j2, j12, m1, m2, -m12);
                    c2 := (-1)^(-j3 + j4 - m34)·sqrt(2·j34 + 1)· W3j(j3, j4, j34, m3, m4, -m34);
                    c3 := (-1)^(-j12 + j34 - M)·sqrt(2·J + 1)· W3j(j12, j34, J, m12, m34, -M);
                    printf("                    %s[%a]", s, SQ(c1·c2·c3) );
                    printf("|%a,%a,%a,%a;%a,%a,%a,%a> \n", j1, j2, j3, j4, m1, m2, m3, m4);
                    s := " + ";
                  end if;
                end do;
              end do;
            end do;
          end do;
        end do;
      end do;
      print( );
    end proc:
```

This procedure returns the product of the CG coefficients and the state |j1,j2,j3,j4;m1,m2,m3,m4⟩

{j1,j2,j12,m1,m2,m12}{j3,j4,j34,m3,m4,m34}{j12,j34,J,m12,m34,M}|j1,j2,j3,j4;m1,m2,m3,m4⟩

where j12 = j1 + j2, j34 = j3 + j4 and J = j12 + j34
m12 = {j12, j12 - 1, ... , − j12} = m1 + m2
m34 = { j34, j34 -1, ... , -j34} = m3 + m4
M = { J, J - 1, ... , − J} = m12 + m34

```
> SCo :=proc(j1, j2, j3, j4, j12, j34, J, M)
        local m1, m2, m3, m4, m12, m34, s;
        s := " = ";
      #   printf("                    =========\n");
        printf("|%a,%a,%a,%a;%a,%a,%a,%a>\n", j1, j2, j3, j4, j12, j34, J, M);
        for  m12 from -j12 by 1 to j12 do
          for m34 from -j34 by 1 to j34 do
            for  m1 from -j1 by 1 to j1 do
              for m2 from -j2 by 1 to  j2 do
                for m3 from -j3 by 1 to j3 do
                  for m4 from -j4 by 1 to j4 do
                    if (m1 + m2) = m12 and (m3 + m4) = m34 and (m12 + m34) = M   then
    # selection rule
                    printf("                    %s{%a,%a,%a,%a,%a,%a}", s, j1, j2, j12, m1, m2, m12 );
                    printf("{%a,%a,%a,%a,%a,%a}", j3, j4, j34, m3, m4, m34);
                    printf("{%a,%a,%a,%a,%a,%a}", j12, j34, J, m12, m34, M);
                    printf("|%a,%a,%a%a;%a,%a,%a,%a> \n", j1, j2, j3, j4, m1, m2, m3, m4);
                    s := " + ";
                  end if;
                end do;
              end do;
            end do;
          end do;
        end do;
        end do;
      end proc:
```

Main procedure. Add4j adds four angular momenta - Add4j(j1,j2,j3,j4)
using the coupling scheme j1 + j2 = j12, j3 + j4 = j34, and j12 + j34 = J

$$|j1 - j2| \leq j12 \leq (j1+j2)$$
$$m12 = \{j12, j12 - 1, ... , -j12\}$$

$$|j4 - j3| \leq j34 \leq (j3 + j4)$$
$$m34 = \{ j34, j34 -1, ... , -j34\}$$

$$|j34 - j12| \leq J \leq (j12 + j34)$$
$$M = \{ J, J - 1, ... , -J\}$$

```
> Add4j := proc(j1, j2, j3, j4, v)
          local c, j12, m12, j34, J, M;
          printf("\n     There are %a |j1,j2,j3,j4;j12,j34,J,M> states where,\n\n", (2 j1 + 1)·(2 j2 + 1)
      ·(2 j3 + 1)·(2 j4 + 1));
          c := 0 :  # counter
          for j12 from (j1 + j2) by -1 to |j1 - j2| do
             for j34 from (j3 + j4) by - 1 to |j3 - j4| do
                for J from (j12 + j34) by -1 to |j12 - j34| do
                   for M from J by -1 to -J do
                      c := c + 1 : printf("%a.", c);
                      if v = 1 then SCo(j1, j2, j3, j4, j12, j34, J, M) end if;
                      VCo(j1, j2, j3, j4, j12, j34, J, M, v);
                   end do;
                end do;
             end do;
          end do;
        end proc:
```

Example:

Add4j(j1,j2,j3,j4,v)
where v={1 or 0} 1 = print CG coefficients & values, 0 = print CG values only

> $Add4j\left(\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0\right);$

```
      There are 16 |j1,j2,j3,j4;j12,j34,J,M> states where,

1.|1/2,1/2,1/2,1/2;1,1,2,2>
                       = [1]|1/2,1/2,1/2,1/2;1/2,1/2,1/2,1/2>

2.|1/2,1/2,1/2,1/2;1,1,2,1>
                       = [1/4]|1/2,1/2,1/2,1/2;-1/2,1/2,1/2,1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;1/2,-1/2,1/2,1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;1/2,1/2,-1/2,1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;1/2,1/2,1/2,-1/2>

3.|1/2,1/2,1/2,1/2;1,1,2,0>
                       = [1/6]|1/2,1/2,1/2,1/2;-1/2,-1/2,1/2,1/2>
                       + [1/6]|1/2,1/2,1/2,1/2;-1/2,1/2,-1/2,1/2>
                       + [1/6]|1/2,1/2,1/2,1/2;-1/2,1/2,1/2,-1/2>
                       + [1/6]|1/2,1/2,1/2,1/2;1/2,-1/2,-1/2,1/2>
                       + [1/6]|1/2,1/2,1/2,1/2;1/2,-1/2,1/2,-1/2>
                       + [1/6]|1/2,1/2,1/2,1/2;1/2,1/2,-1/2,-1/2>

4.|1/2,1/2,1/2,1/2;1,1,2,-1>
                       = [1/4]|1/2,1/2,1/2,1/2;-1/2,-1/2,-1/2,1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;-1/2,-1/2,1/2,-1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;-1/2,1/2,-1/2,-1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;1/2,-1/2,-1/2,-1/2>

5.|1/2,1/2,1/2,1/2;1,1,2,-2>
                       = [1]|1/2,1/2,1/2,1/2;-1/2,-1/2,-1/2,-1/2>

6.|1/2,1/2,1/2,1/2;1,1,1,1>
                       = [-1/4]|1/2,1/2,1/2,1/2;-1/2,1/2,1/2,1/2>
                       + [-1/4]|1/2,1/2,1/2,1/2;1/2,-1/2,1/2,1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;1/2,1/2,-1/2,1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;1/2,1/2,1/2,-1/2>

7.|1/2,1/2,1/2,1/2;1,1,1,0>
                       = [-1/2]|1/2,1/2,1/2,1/2;-1/2,-1/2,1/2,1/2>
                       + [0]|1/2,1/2,1/2,1/2;-1/2,1/2,-1/2,1/2>
                       + [0]|1/2,1/2,1/2,1/2;-1/2,1/2,1/2,-1/2>
                       + [0]|1/2,1/2,1/2,1/2;1/2,-1/2,-1/2,1/2>
                       + [0]|1/2,1/2,1/2,1/2;1/2,-1/2,1/2,-1/2>
                       + [1/2]|1/2,1/2,1/2,1/2;1/2,1/2,-1/2,-1/2>
```

```
8.|1/2,1/2,1/2,1/2;1,1,1,-1>
                       = [-1/4]|1/2,1/2,1/2,1/2;-1/2,-1/2,-1/2,1/2>
                       + [-1/4]|1/2,1/2,1/2,1/2;-1/2,-1/2,1/2,-1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;-1/2,1/2,-1/2,-1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;1/2,-1/2,-1/2,-1/2>

9.|1/2,1/2,1/2,1/2;1,1,0,0>
                       = [1/3]|1/2,1/2,1/2,1/2;-1/2,-1/2,1/2,1/2>
                       + [-1/12]|1/2,1/2,1/2,1/2;-1/2,1/2,-1/2,1/2>
                       + [-1/12]|1/2,1/2,1/2,1/2;-1/2,1/2,1/2,-1/2>
                       + [-1/12]|1/2,1/2,1/2,1/2;1/2,-1/2,-1/2,1/2>
                       + [-1/12]|1/2,1/2,1/2,1/2;1/2,-1/2,1/2,-1/2>
                       + [1/3]|1/2,1/2,1/2,1/2;1/2,1/2,-1/2,-1/2>

10.|1/2,1/2,1/2,1/2;1,0,1,1>
                       = [-1/2]|1/2,1/2,1/2,1/2;1/2,1/2,-1/2,1/2>
                       + [1/2]|1/2,1/2,1/2,1/2;1/2,1/2,1/2,-1/2>

11.|1/2,1/2,1/2,1/2;1,0,1,0>
                       = [-1/4]|1/2,1/2,1/2,1/2;-1/2,1/2,-1/2,1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;-1/2,1/2,1/2,-1/2>
                       + [-1/4]|1/2,1/2,1/2,1/2;1/2,-1/2,-1/2,1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;1/2,-1/2,1/2,-1/2>

12.|1/2,1/2,1/2,1/2;1,0,1,-1>
                       = [-1/2]|1/2,1/2,1/2,1/2;-1/2,-1/2,-1/2,1/2>
                       + [1/2]|1/2,1/2,1/2,1/2;-1/2,-1/2,1/2,-1/2>

13.|1/2,1/2,1/2,1/2;0,1,1,1>
                       = [-1/2]|1/2,1/2,1/2,1/2;-1/2,1/2,1/2,1/2>
                       + [1/2]|1/2,1/2,1/2,1/2;1/2,-1/2,1/2,1/2>

14.|1/2,1/2,1/2,1/2;0,1,1,0>
                       = [-1/4]|1/2,1/2,1/2,1/2;-1/2,1/2,-1/2,1/2>
                       + [-1/4]|1/2,1/2,1/2,1/2;-1/2,1/2,1/2,-1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;1/2,-1/2,-1/2,1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;1/2,-1/2,1/2,-1/2>
```

```
15.|1/2,1/2,1/2,1/2;0,1,1,-1>
                       = [-1/2]|1/2,1/2,1/2,1/2;-1/2,1/2,-1/2,-1/2>
                       + [1/2]|1/2,1/2,1/2,1/2;1/2,-1/2,-1/2,-1/2>

16.|1/2,1/2,1/2,1/2;0,0,0,0>
                       = [1/4]|1/2,1/2,1/2,1/2;-1/2,1/2,-1/2,1/2>
                       + [-1/4]|1/2,1/2,1/2,1/2;-1/2,1/2,1/2,-1/2>
                       + [-1/4]|1/2,1/2,1/2,1/2;1/2,-1/2,-1/2,1/2>
                       + [1/4]|1/2,1/2,1/2,1/2;1/2,-1/2,1/2,-1/2>
```

(1)

```
>
```