> *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 Two 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, C-G, coefficients 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, j2 - j1 - m3);
        kmax := min( j3 - j1 + j2, j3 - m3);

        P := (-1)^(j1 - j2 - m3);

        N := sqrt( (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)!) );

        S := sum( (-1)^(k + j2 + m2)·(j2 + j3 + m1 - k)!(j1 - m1 + k)! / ((k)!(j3 - j1 + j2 - k)!(j3 - m3 - k)!(k + j1 - j2 + m3)!), k = kmin..kmax );

        return (P·N·S);

    end proc:
```

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

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

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

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

⟨j1,j2;m1,m2|j1,j2;j12,m12⟩ = [± N]

[± N] ≡ ± $\sqrt{\mathbf{N}}$

```
> VCo := proc(j1, j2, j12, m12)
        local m1, m2, c, s, sign;
        s := " = ";
        for m1 from j1 by -1 to -j1 do
          for m2 from j2 by -1 to -j2 do
            if (m1 + m2) = m12  then  # selection rule
              c := (-1)^(-j1 + j2 - m12)·sqrt(2·j12 + 1) · W3j( j1, j2, j12, m1, m2, -m12);
              if (evalf(c) < 0) then sign := -1 else sign := 1 end if;
              c := sign·simplify(c·c);
              printf("                %s[%a]", s, c);
              printf("|%a,%a;%a,%a> \n", j1, j2, m1, m2);
             s := " + ";
            end if;
          end do;
        end do;
      end proc:
```

This procedure returns the CG coefficients ⟨j1,j2;m1,m2|j1,j2;j12,m12⟩ and the state |j1,j2;m1,m2⟩

⟨j1,j2;m1,m2|j1,j2;j12,m12⟩|j1,j2;m1,m2⟩

where j12 = j1 + j2
m12 = m1 + m2

```
> SCo :=proc(j1, j2, j12, m12)
        local m1, m2, c, s;
        s := " = ";
        printf("\|%a,%a;%a,%a>\n", j1, j2, j12, m12);
        for m1 from j1 by -1 to -j1 do
          for m2 from j2 by -1 to -j2 do
            if (m1 + m2) = m12 then   # selection rule
              printf("          %s<%a,%a;%a,%a|%a,%a;%a,%a>", s, j1, j2, m1, m2, j1, j2, j12,
   m12 );
              printf("|%a,%a;%a,%a> \n", j1, j2, m1, m2);
              s := " + ";
            end if;
          end do;
        end do;
      end proc:
```

Main procedure. Add2j adds two angular momenta j1, j2 - Add2j(j1,j2)

|j1 - j2| ≤ j12 ≤ j1 + j2

m12 = { j12, j12 - 1, j12 - 2, ... , -j12}

```
> Add2j := proc(j1, j2)
        local c, j12, m12;
        printf("\n   There are %a |j1,j2;j12,m12> states where,\n", (2 j1 + 1)·(2 j2 + 1));
        print(" |j1,j2;j12,m12> = Σ ⟨j1, j2; m1, m2|j1, j2; j12, m12⟩|j1, j2; m1, m2 > ");
        c := 0;
        for j12 from (j1 + j2) by -1 to |j1 - j2| do
          for m12 from j12 by -1 to -j12 do
            c := c + 1 : printf("%a.", c);
            SCo(j1, j2, j12, m12);
            VCo(j1, j2, j12, m12);
          end do;
        end do;

     end proc:
```

Examples Add2j(j1,j2)

```
> Add2j(1/2, 1/2);

     There are 4 |j1,j2;j12,m12> states where,
                  " |j1,j2;j12,m12> = Σ <j1,j2;m1,m2|j1,j2;j12,m12>|j1,j2;m1,m2> "
1.|1/2,1/2;1,1>
                 = <1/2,1/2;1/2,1/2|1/2,1/2;1,1>|1/2,1/2;1/2,1/2>
                 = [1]|1/2,1/2;1/2,1/2>
2.|1/2,1/2;1,0>
                 = <1/2,1/2;1/2,-1/2|1/2,1/2;1,0>|1/2,1/2;1/2,-1/2>
                 + <1/2,1/2;-1/2,1/2|1/2,1/2;1,0>|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>
3.|1/2,1/2;1,-1>
                 = <1/2,1/2;-1/2,-1/2|1/2,1/2;1,-1>|1/2,1/2;-1/2,-1/2>
                 = [1]|1/2,1/2;-1/2,-1/2>

4.|1/2,1/2;0,0>
                 = <1/2,1/2;1/2,-1/2|1/2,1/2;0,0>|1/2,1/2;1/2,-1/2>
                 + <1/2,1/2;-1/2,1/2|1/2,1/2;0,0>|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>
> Add2j(1, 1);

     There are 9 |j1,j2;j12,m12> states where,
                  " |j1,j2;j12,m12> = Σ <j1,j2;m1,m2|j1,j2;j12,m12>|j1,j2;m1,m2> "
1.|1,1;2,2>
                 = <1,1;1,1|1,1;2,2>|1,1;1,1>
                 = [1]|1,1;1,1>
2.|1,1;2,1>
                 = <1,1;1,0|1,1;2,1>|1,1;1,0>
                 + <1,1;0,1|1,1;2,1>|1,1;0,1>
                 = [1/2]|1,1;1,0>
                 + [1/2]|1,1;0,1>
3.|1,1;2,0>
                 = <1,1;1,-1|1,1;2,0>|1,1;1,-1>
                 + <1,1;0,0|1,1;2,0>|1,1;0,0>
                 + <1,1;-1,1|1,1;2,0>|1,1;-1,1>
                 = [1/6]|1,1;1,-1>
                 + [2/3]|1,1;0,0>
                 + [1/6]|1,1;-1,1>
```

```
4.|1,1;2,-1>
                 = <1,1;0,-1|1,1;2,-1>|1,1;0,-1>
                 + <1,1;-1,0|1,1;2,-1>|1,1;-1,0>
                 = [1/2]|1,1;0,-1>
                 + [1/2]|1,1;-1,0>
5.|1,1;2,-2>
                 = <1,1;-1,-1|1,1;2,-2>|1,1;-1,-1>
                 = [1]|1,1;-1,-1>
6.|1,1;1,1>
                 = <1,1;1,0|1,1;1,1>|1,1;1,0>
                 + <1,1;0,1|1,1;1,1>|1,1;0,1>
                 = [1/2]|1,1;1,0>
                 + [-1/2]|1,1;0,1>
7.|1,1;1,0>
                 = <1,1;1,-1|1,1;1,0>|1,1;1,-1>
                 + <1,1;0,0|1,1;1,0>|1,1;0,0>
                 + <1,1;-1,1|1,1;1,0>|1,1;-1,1>
                 = [1/2]|1,1;1,-1>
                 + [0]|1,1;0,0>
                 + [-1/2]|1,1;-1,1>
8.|1,1;1,-1>
                 = <1,1;0,-1|1,1;1,-1>|1,1;0,-1>
                 + <1,1;-1,0|1,1;1,-1>|1,1;-1,0>
                 = [1/2]|1,1;0,-1>
                 + [-1/2]|1,1;-1,0>
9.|1,1;0,0>
                 = <1,1;1,-1|1,1;0,0>|1,1;1,-1>
                 + <1,1;0,0|1,1;0,0>|1,1;0,0>
                 + <1,1;-1,1|1,1;0,0>|1,1;-1,1>
                 = [1/3]|1,1;1,-1>
                 + [-1/3]|1,1;0,0>
                 + [1/3]|1,1;-1,1>
>
```