```cs
using System;

namespace InteractionEnergies
{
    public class RadialWaveFunction
    {
        // SI Units from NIST website

        public double alpha = 1.0 / 137.0;
        public double c = 299792458;
        public double h = 6.62607015e-34;
        public double hbar = 1.054571817e-34;
        public double me = 9.1093837015e-31;
        public double mn = 1.67492749804e-27;
        public double mp = 1.67262192369e-27;
        public double e = 1.602176634e-19;
        public double e0 = 8.8541878128e-12;

        // constants are updated from Schiff's
        // cgs system numbers found in the
        // first page of his "Quantum Mechanics"

        public double e_schiff = 4.806529902e-10;
        public double hbar_schiff = 1.054571817e-27;
        public double me_schiff = 0.91093837015e-27;
        public double mn_schiff = 1.674927498045e-24;
        public double mp_schiff = 1.67262192369e-24;

        public double a_mu_schiff_unscaled(double mu)
        {
            return hbar_schiff * hbar_schiff / (mu * e_schiff * e_schiff);
        }

        public double a_mu_schiff_scaled(double mu)
        {
            return 1.0e8 * a_mu_schiff_unscaled(mu);
        }

        public double En_schiff_unscaled(double amu, int Z, int n)
        {
            return -Z * Z * e_schiff * e_schiff
                / (2.0 * amu * n * n);
        }

        public double En_schiff_scaled(double amu, int Z, int n)
        {
            return 6.24150907446076e11 * En_schiff_unscaled(amu, Z, n);
        }

        public double a_mu_SI_Units_unscaled(double mu, int Z)
        {
            return 4.0 * Math.PI * e0 * hbar * hbar / (mu * Z * e * e);
```

```
53          }
54
55          public double a_mu_SI_Units_scaled(double mu, int Z)
56          {
57              return 1e-10 * a_mu_SI_Units_unscaled(mu, Z);
58          }
59
60          public double En_SI_Units_unscaled(double amu, double mu, int Z, int n)
61          {
62              return -Z * Z * mu * e * e * e * e /
63                  (8.0 * n * n * h * h * e0 * e0);
64          }
65
66          public double En_SI_Units_scaled(double amu, double mu, int Z, int n)
67          {
68              return 6.24150907446076e18 * En_SI_Units_unscaled(amu, mu, Z, n);
69          }
70
71          public double Mu(double m, double M)
72          {
73              return (m * M) / (m + M);
74          }
75
76          public double Factorial(int n)
77          {
78              double nfact = 1.0;
79
80              for (int i = 2; i <= n; i++)
81                  nfact *= i;
82
83              return nfact;
84          }
85
86          // Associated Laguerre Functions
87          // https://mathworld.wolfram.com/AssociatedLaguerrePolynomial.html
88
89          public double Laguerre(double rho, int n, int l)
90          {
91              double sum = 0.0;
92
93              for (int k = 0; k <= n - l - 1; k++)
94              {
95                  double factor1 = Math.Pow(-1, k + 2 * l + 1);
96                  double factor2 = Factorial(n + l);
97                  double factor3 = Math.Pow(rho, k);
98                  double numer = factor1 * factor2 * factor2 * factor3;
99                  double factor4 = Factorial(n - l - 1 - k);
100                 double factor5 = Factorial(2 * l + 1 + k);
101                 double factor6 = Factorial(k);
102                 double denom = factor4 * factor5 * factor6;
103
104                 sum += numer / denom;
```

```
105             }
106
107             return sum;
108         }
109
110         public double Rnl(double amu, double r, int n, int l, int Z)
111         {
112             double factor1 = 2.0 * Z / (n * amu);
113             double rho = factor1 * r;
114             double factor2 = Math.Pow(factor1, 3.0);
115             double numer = Factorial(n - l - 1);
116             double denom = 2.0 * n * Math.Pow(Factorial(n + l), 3.0);
117             double factor3 = Math.Sqrt(factor2 * numer / denom);
118             double factor4 = Math.Exp(-0.5 * rho) * Math.Pow(rho, l);
119             double factor5 = Laguerre(rho, n, l);
120
121             factor5 = -factor3 * factor4 * factor5;
122
123             return factor5;
124         }
125     }
126 }
```