```cpp
/*
 * Integrationsroutinen.cpp
 *
 *  Created on: 24.04.2017
 *      Author: mona
 */



#include <iostream>
#include <complex>
#include <cstdlib>
#include <vector>
#include <cmath>
#include <sstream>
#include <utility>
#include <math.h>
#include <fstream>
#include <functional>
using namespace std;


double F1(double x){
        if(x==0){
                        x=pow(10, -9);
                }

        double f1=exp(-x)/x;
        return f1;
}

double F2(double x){
        if(x==0){
                return 0;
        }

        double f2=x*sin(1/x);
        return f2;
}

double trapez(double a, double b, double N, double(*f)(double)){
        double h=(b-a)/N;
        double summe=(h/2)*(f(a)+f(b));
        for (int n=1; n<N; n++){
                summe=summe+h*f(a+n*h);
        };
        return summe;
}

double mittelpunkt(double a, double b, double N, double(*f)(double)){
        double h=(b-a)/N;
        double summe=0;
                for (int n=0; n<N; n++){
                        summe=summe+f(a+(h/2)+(n*h));
                };
        return h*summe;
}


double simpson(double a, double b, double N, double(*f)(double)){
        double h=(b-a)/N;
        double summe= f(a)+f(b);
        for (int n=1; n<N; n++){
                                if(n%2==0){summe=summe+2*f(a+h*n);}
                                else{summe=summe+4*f(a+h*n);}
        }
        return (h/3)*summe;
}


double eps=pow(10, -4);

int main(){

        int a1=1;
        int b1=100;
        int a2=0;
        int b2=1;
        double Delta = 1;
        double N=2;
        double links;
        double rechts;

        //Abspeichern der Daten I1 mit Trapezregel
        ofstream a;
        a.open("I1_Trapez.txt");
        a.precision(10);
        //

        while(Delta>eps){
                links=trapez(a1,b1,N/2,F1);
                rechts=trapez(a1,b1,N,F1);
                Delta=abs(rechts-links)/links;
                a << N/2 << "\t" << links <<  "\n";
                N=2*N;
        }
        a.close();


        //Abspeichern der Daten I2 mit Trapezregel
        ofstream b;
        b.open("I2_Trapez.txt");
        b.precision(10);
        //

        N=2;
        Delta=1;

        while(Delta>eps){
                links=trapez(a2,b2,N/2,F2);
                rechts=trapez(a2,b2,N,F2);
                Delta=abs(rechts-links)/links;
                b << N/2 << "\t" << links <<  "\n";
                N=2*N;
        }
        b.close();

        //Abspeichern der Daten I1 mit Mittelpunktsregel
        ofstream c;
        c.open("I1_Mittelpunkt.txt");
        c.precision(10);
        //

        N=2;
        Delta=1;

        while(Delta>eps){
                links=mittelpunkt(a1,b1,N/2,F1);
                rechts=mittelpunkt(a1,b1,N,F1);
                Delta=abs(rechts-links)/links;
                c << N/2 << "\t" << links <<  "\n";
                N=2*N;
        }
        c.close();


        //Abspeichern der Daten I2 mit Mittelpunktsregel
        ofstream d;
        d.open("I2_Mittelpunkt.txt");
        d.precision(10);
        //

        N=2;
        Delta=1;

        while(Delta>eps){
                links=mittelpunkt(a2,b2,N/2,F2);
                rechts=mittelpunkt(a2,b2,N,F2);
                Delta=abs(rechts-links)/links;
                d << N/2 << "\t" << links <<  "\n";
                N=2*N;
        }
        d.close();

        //Abspeichern der Daten I1 mit Simpsonregel
        ofstream g;
        g.open("I1_Simpson.txt");
        g.precision(10);
        //

        N=2;
        Delta=1;

        while(Delta>eps){
                links=simpson(a1,b1,N/2,F1);
                rechts=simpson(a1,b1,N,F1);
                Delta=abs(rechts-links)/links;
                g << N/2 << "\t" << links <<  "\n";
                N=2*N;
        }
        g.close();

```

```cpp

        //Abspeichern der Daten I2 mit Simpson
        ofstream k;
        k.open("I2_Simpson.txt");
        k.precision(10);
        //

        N=2;
        Delta=1;

        while(Delta>eps){
                links=simpson(a2,b2,N/2,F2);
                rechts=simpson(a2,b2,N,F2);
                Delta=abs(rechts-links)/links;
                k << N/2 << "\t" << links <<  "\n";
                N=2*N;
        }
        k.close();


        return 0;
}
```

```cpp
/*
 * Punktladungen.cpp
 *
 *  Created on: 24.04.2017
 *      Author: mona
 */



#include <iostream>
#include <complex>
#include <cstdlib>
#include <vector>
#include <cmath>
#include <sstream>
#include <utility>
#include <math.h>
#include <fstream>
#include <functional>
using namespace std;

double Energie(double r, double a){
        double r1=sqrt(2*a*a-2*a*r+r*r);
        double r2=sqrt(2*a*a+2*a*r+r*r);
        double E=-2*((1/r1)+(1/r2));
        return E;
}

double zPunkt(double h, double r, double a){
        double f=Energie(r+h,a)-Energie(r-h,a);
        f=f/(2*h);
        return f;
}

double Kraft(double r, double a){
        double t1=sqrt(2*a*a-2*a*r+r*r)*(2*a*a-2*a*r+r*r); //entspricht ^3/2
        double t2=sqrt(2*a*a+2*a*r+r*r)*(2*a*a+2*a*r+r*r);
        t1=(-2*a+2*r)/t1;
        t2=(2*a+2*r)/t2;
        double F=t1+t2;
        return -F;
}

int main(){
        double a=1;
        double relFehler;
        //Abspeichern der Daten fÃ¼r die Energie
        ofstream b;
        b.open("Energie.txt");
        b.precision(10);
        //

        for(double n=-3*a; n<=3*a; n=n+0.001){
                b << n << "\t" << Energie(n,a) <<  "\n";
        }
        b.close();

        //Abspeichern der Daten
                ofstream c;
                c.open("Kraft_03a.txt");
                c.precision(10);
                //

                for(double n=-3*a; n<=3*a; n=n+0.001){
                        relFehler=(-zPunkt(0.3*a,n,a)-Kraft(n,a))/Kraft(n,a);
                        c << n << "\t" << -zPunkt(0.3*a,n,a) << "\t" << Kraft(n,a) << "\t" << relFehler <<"\n";
                }
                c.close();

                //Abspeichern der Daten
                ofstream d;
                d.open("Kraft_E4a.txt");
                d.precision(10);
                //
                double h=pow(10,-4);
                for(double n=-3*a; n<=3*a; n=n+0.001){
                        relFehler=(-zPunkt(h*a,n,a)-Kraft(n,a))/Kraft(n,a);
                        d << n << "\t" << -zPunkt(h*a,n,a) << "\t" << Kraft(n,a)<< "\t" << relFehler  << "\n";
                }
                d.close();

                //Abspeichern der Daten
                ofstream g;
                g.open("Kraft_E15a.txt");
                g.precision(10);
                                //
                h=pow(10,-15);
                for(double n=-3*a; n<=3*a; n=n+0.001){
                        relFehler=(-zPunkt(h*a,n,a)-Kraft(n,a))/Kraft(n,a);
                        g << n << "\t" << -zPunkt(h*a,n,a) << "\t" << Kraft(n,a) << "\t" << relFehler << "\n";
                                }
                g.close();


        return 0;
}

```